Quantum one-time programs

Anne Broadbent[†] Gus Gutoski[‡] Douglas Stebila[*]

[†]*Institute for Quantum Computing and Department of Combinatorics and Optimization*
University of Waterloo, Waterloo, Ontario, Canada
albroadb@iqc.ca

[‡]*Institute for Quantum Computing and School of Computer Science*
University of Waterloo, Waterloo, Ontario, Canada
gus.gutoski@uwaterloo.ca

[*]*School of Electrical Engineering and Computer Science, Science and Engineering Faculty*
Queensland University of Technology, Brisbane, Queensland, Australia
stebila@qut.edu.au

November 6, 2012

Abstract

One-time programs are modelled after a black box that allows a single evaluation of a function, and then self-destructs. Because software can, in principle, be copied, general one-time programs exists only in the hardware token model: it has been shown that any function admits a one-time program as long as we assume access to physical devices called *one-time memories*. Quantum information, with its well-known property of *no-cloning*, would, at first glance, prevent the basic copying attack for classical programs. We show that this intuition is false: one-time programs for both classical and quantum maps, based solely on quantum information, do not exist, even with computational assumptions. We complement this strong impossibility proof by an equally strong possibility result: assuming the same basic one-time memories as used for classical one-time programs, we show that every quantum map has a quantum one-time program that is secure in the universal composability framework. Our construction relies on a new, simpler quantum authentication scheme and corresponding mechanism for computing on authenticated data.

Contents

1 Introduction

A one-time program for a function f, as introduced by Goldwasser, Rothblum and Kalai [GKR08], is a cryptographic primitive by which a receiver may evaluate f on only one input, chosen by the receiver at run time: no efficient adversary, after evaluating the one-time program on x, should be able to learn anything about $f(y)$ for any $y \neq x$ beyond what can be inferred from $f(x)$. Secure one-time programs could have far reaching implications in software protection, digital rights management, and electronic cash or token schemes. (For example, coins are a program that can only be run once, and thus cannot be double spent.)

One-time programs cannot be achieved by software alone, as any software can be copied and executed multiple times. Thus, any hope of achieving any one-time property must necessarily rely on an additional assumptions such as secure hardware or interaction; in particular, computational assumptions alone will not suffice.

1.1 Classical one-time programs from one-time memories

Goldwasser *et al.* showed how to construct a one-time program for any function f using a very basic hypothetical hardware device called a *one-time memory (OTM)*. Inspired by the interactive cryptographic primitive *oblivious transfer*, each OTM stores two secret strings (s_0, s_1). A receiver requests one of these two strings by specifying a single-bit input $c \in \{0, 1\}$. The OTM reveals s_c and then self-destructs: the other string $s_{\overline{c}}$ is lost forever.

One advantage of using OTMs as a building block is their simplicity: an OTM is an extremely basic device that does not perform any computation. Using the simplest possible hardware device allows for easier scrutiny against potential hardware flaws such as side-channel attacks. Moreover, the functionality of an OTM is independent of the program itself, and thus OTMs could be mass-produced for a variety of programs. The use of tamper-proof hardware in cryptography is an old and recurring theme [Smi81], and OTMs in particular have lead to a recent revival in ascertaining what cryptographic primitives can be constructed using minimalistic hardware assumptions that could not otherwise be achieved.

Non-interactive secure two-party computation. Goyal, Ishai, Sahai, Venkatesan, and Wadia [GIS+10] improved on the work of Goldwasser *et al.* [GKR08] in several ways. First, they consider a more general primitive, which they call *non-interactive secure two-party computation*, in which two parties wish to evaluate a publicly known function $f(x, y)$. One party—the *sender*—is given the input string x. The sender uses x to prepare a "program" $p(x)$ and sends this program to the *receiver*. The receiver wishes to use the program $p(x)$ in order to evaluate $f(x, y)$ for any input string y of her choice. Like one-time programs, after evaluating $f(x, y)$, no adversary should be able to learn anything about $f(x, y')$ for any $y' \neq y$ beyond what can be inferred from $f(x, y)$. The one-time programs of Goldwasser *et al.* are recovered as a special case of this primitive by viewing the input x as a description of a function $g_x(y)$ and the publicly known function f as a "universal computer" that produces $f(x, y) = g_x(y)$. Non-interactive secure two-party computation is impossible in the plain model for the same reason that one-time programs are impossible in the plain model: software can always be copied.

Second, the one-time programs of Goldwasser *et al.* are secure against a malicious receiver; the issue of a malicious sender does not arise in their setting. By contrast, in the more general setting of non-interactive secure two-party computation one could also consider malicious *senders*. The protocol of Goyal *et al.* is secure against both a malicious receiver *and* a malicious sender.

Third, Goldwasser *et al.* use OTMs for large strings, whereas Goyal *et al.* require OTMs for only *single bits*. Finally, Goldwasser *et al.* establish security against computationally bounded adversaries, whereas Goyal *et al.* establish statistical universally composable security.

Terminology: For brevity, we use the term *one-time program (OTP)* synonymously with "non-interactive secure two-party computation".

1.2 Impossibility of quantum one-time programs in the plain model

In contrast to ordinary classical information, quantum information cannot in general be copied: measurement is an irreversible, destructive process [WZ82]. The no-cloning property of quantum information is credited for such classically impossible feats as quantum money [AC12, MS10, Wie83], quantum key distribution [BB84], and quantum copy-protection [Aar09]. It is thus natural to ask if one-time programs can be added to this list of quantum cryptographic primitives: *does quantum information allow for one-time programs without hardware assumptions?* (When there are no hardware assumptions, we refer to this as the *plain quantum model*.)

We observe in Section 3 that the answer to this question is a strong *no*: although quantum information cannot be copied, a quantum "program state" for f can always be re-constructed by a reversible adversary after each use so as to obtain the evaluation of f on multilple distinct inputs. In particular, computational assumptions do not help to achieve quantum one-time programs.

One-time programs for quantum channels. By analogy to classical functions acting on bits, one could also consider a one-time program for a quantum channel $\Phi : (\mathsf{A}, \mathsf{B}) \to \mathsf{C}$ acting on multi-qubit registers A (the sender's input), B (the receiver's input), and C (the receiver's output). The security goal is similar in spirit to that for classical functions: for each joint state ρ of the input registers (A, B) no adversary should be able to learn anything about $\Phi(\rho')$ for any state $\rho' \neq \rho$ beyond what can be inferred from $\Phi(\rho)$.

Here, again, our previous observation on the impossibility of quantum one-time programs in the plain model holds: if a quantum program allows the adversary to evaluate $\Phi(\rho)$ then that same program can be recovered by a reversible adversary in order to subsequently evaluate $\Phi(\rho')$ for any ρ' that can be obtained from ρ by a local operation on B.

1.3 Main result: quantum one-time programs from one-time memories

Given that one-time programs do not exist for arbitrary quantum channels in the plain quantum model, and that one-time programs do exist for arbitrary classical functions in the OTM model, a natural question arises: *what additional assumptions are required to achieve one-time programs for quantum channels?*

Our main result (Theorem 7) is that any channel Φ can be compiled into a *quantum one-time program (QOTP)*, which is a combination of quantum states and OTMs that allows a receiver to evaluate Φ exactly once. In particular, (and perhaps surprisingly) single-bit *classical* one-time memory devices suffice to establish *quantum* one-time programs for arbitrary *quantum* channels. An informal version of Theorem 7 is as follows:

Main theorem (informal). *For each channel Φ specified by a quantum circuit, there is a non-interactive two-party protocol for the secure evaluation of Φ, assuming classical one-time memory devices and an honest sender. Moreover, this protocol is universally composable (UC-secure).*

We provide a fully rigorous proof of statistical universally composable (UC) security of our

QOTPs. The question of QOTPs that are also secure against a malicious sender is left for future work. For simplicity, we restrict our attention to the case of *non-reactive* one-time quantum programs. The more general scenario of being able to query a program a bounded number of times (including the case of an n-use program) may be implemented using standard techniques as is done in the classical case (see Section 2.3). Most of the components of our QOTP for Φ are independent of the sender's input register A and so can be compiled (and mass-produced) by the sender before he receives his input. As a corollary of our main result we obtain the UC security of the protocol for *delegated quantum computations* of Aharonov, Ben-Or and Eban [ABOE10]; see Section 7.

Summary of techniques. Our protocol employs a method for *quantum computing on authenticated data (QCAD)*, which allows for the application of quantum gates to authenticated quantum data without knowing the authentication key. We propose a new authentication scheme, called the *trap scheme*, and show that it allows for QCAD (Section 4.1). Prior to our work, the only authentication scheme known to admit QCAD was the *signed polynomial scheme* of Ben-Or, Crépeau, Gottesman, Hassidim, and Smith [BOCG$^+$06] (see also [ABOE10]). We compare our trap scheme to the signed polynomial scheme in Section 4.2. Recently, and independently of our work, it was shown by Dupuis, Nielsen, and Salvail [DNS12] that the *Clifford authentication scheme* also admits QCAD.

In methods for QCAD, universal quantum computation can only be performed if the receiver (who holds the authenticated data) is allowed to exchange classical messages with the sender (who knows the authentication key). To keep our protocol non-interactive, all the classical interaction is encapsulated by a *bounded, reactive classical one-time program (BR-OTP)* prepared by the sender. (The existence of secure reactive one-time programs follows from the work of [GIS$^+$10] as described in Section 2.3.) This program for the BR-OTP depends upon the authentication key chosen for the sender's input register, but *not* on the contents of that register. Thus, by selecting this key in advance, the BR-OTP can be prepared (or mass-produced) before the sender gets his input register.

In order to implement QCAD, the receiver's input must be authenticated prior to computation. This is accomplished non-interactively by having the sender prepare a pair of registers in a special "teleport-through-encode" state, which is a maximally entangled state with an authentication operation applied to one of the two registers. The authentication key is determined by the (classical) result of the Bell measurement used for teleportation. The sender allows the receiver to non-interactively de-authenticate the output at the end of the computation by preparing another pair of registers in a "teleport-through-decode" state. In order to successfully de-authenticate, the receiver's messages to the BR-OTP must be consistent with the secret authentication key held by the BR-OTP. Otherwise, the BR-OTP simply declines to reveal the final decryption key for the receiver's output.

1.4 Formalizing impossibility

In preparing a formal proof that one-time programs do not exist in the plain model, one immediately encounters a pathological class of functions that do, in fact, admit classical one-time programs in the plain model. For example, consider the function $f(a, b) = a+b$. A potential "one-time" program for f has the sender simply reveal a to the receiver. Curiously, this is indeed a "one-time" program, because this behaviour can be simulated with one-shot access to the ideal functionality $f(a, \cdot)$: the query $f(a, 0)$ reveals a, which is exactly the one-time program prepared by the sender. Put another way: f is a function for which there exists a one-time program in the plain model, but only because the one-time program reveals enough information that even a simulator with one-shot access to

$f(a, \cdot)$ can gain all information required to compute the function. This phenomenon is somewhat akin to trivially obfuscatable functions [BGI+01].

An interesting question thus arises: *what is the class of classical functions or quantum channels that admit one-time programs in the plain model?* We provide a complete characterization of this class for classical functions. In particular, we define a class of *unlockable* functions consisting of those functions f for which there exists at least one *key input* b_0 such that for all a the value $f(a, b)$ can be recovered from $f(a, b_0)$ for any desired b. We prove the following.

Impossibility theorem for classical functions (informal). *If f is unlockable then f admits a trivial classical one-time program in the plain model. Conversely, if f is not unlockable then f does not admit a quantum one-time program in the plain quantum model.*

The situation for quantum channels is very interesting. We propose two definitions for the unlockability of a channel, which we call *weakly unlockable* and *strongly unlockable* (Definition 3). We prove the following.

Impossibility theorem for quantum channels (informal). *If Φ is strongly unlockable then Φ admits a trivial quantum one-time program in the plain quantum model. Conversely, if Φ is not weakly unlockable then Φ does not admit a quantum one-time program in the plain quantum model.*

By definition, every strongly unlockable channel is also weakly unlockable. We conjecture that the converse also holds (Conjecture 6.) In lieu of a full proof, we provide a high-level outline of what such a proof might look like in Section 3.4. It appears, in asking a question about unlockable channels, that we have stumbled upon a deep and interesting question relating to the invertible subspaces of an arbitrary channel, akin to the so-called "decoherence-free" subspaces studied in the literature on quantum error correction.

1.5 Related work

Copy-protection. In software copy-protection [Aar09], a program can be evaluated (a possibly unlimited number of times), but it should be impossible for the program to be "split" or "copied" into parts allowing separate executions. As with OTPs, copy-protection cannot be achieved by software means only. OTPs provide a hardware solution by enforcing that the program be run only once. However, the more interesting question is if quantum information alone (with computational assumptions) can provide a solution. Aaronson [Aar09] has proposed such schemes based on plausible, but non-standard, cryptographic assumptions. It is an open problem if quantum copy-protection could be based on standard assumptions. In contrast, the security of quantum OTPs is based on simple OTMs; it could be beneficial to study quantum copy-protection in light of our result.

Quantum money. Our construction establishes quantum authentication codes (see Section 4.1) that seem to provide a concrete and efficient realization of the "hidden subspaces" used for public-key quantum money scheme of Aaronson and Christiano [AC12]. Our QOTPs can also be used to implement non-interactive verification for quantum coin schemes [MS10].

Program obfuscation. A related but different task is program obfuscation, in which the receiver should not be able to efficiently "learn" anything from the description of the program that he could not also efficiently learn from the input-output behaviour of the program. In the case of

classical information, it is known that secure program obfuscation is not possible in the plain model [BGI+01]. As with copy-protection, OTPs provide a hardware solution by enforcing that the obfuscated program can be run only a limited number of times. Again, the more interesting question is if quantum information alone (with computational assumptions) can provide a solution; the impossibility proof for obfuscation breaks down in the quantum case due to the no-cloning theorem. It is an open problem whether there is a way to substitute the assumption of secure hardware in our main possibility result with a computational assumption in order to realize quantum program obfuscation.

2 Notation and tools

A *quantum register* is a specific quantum system (composed, say, out of qubits). Quantum registers are typically denoted with sans serifs such as A, B, *etc.* A *quantum channel* $\Phi : \mathsf{A} \to \mathsf{B}$ refers to any physically-realizable mapping on quantum registers. We use the terms *quantum map* and *quantum channel* interchangeably.

2.1 Universal composability

The universal composability (UC) framework provides an extremely high standard for establishing a strong and rigorous notion of security. The basic idea is to postulate an ideal world, where the protocol parties interact with an *ideal functionality*, which is secure by definition. Then, we consider the real world, where the protocol parties execute the actual protocol. *UC-security* holds if, for every real-world adversary, there exists a *simulator* in the ideal world (taking the role of the real-world adversary) such that no *environment* can distinguish the real and ideal worlds. The environment is powerful: it provides the parties' inputs, receives outputs and *interacts* with the adversary at arbitrary points in the protocol. *Perfect*, *statistical*, and *computational* notions of UC security exist depending on the two worlds being equal or statistically or computationally indistinguishable.

The UC framework was developed in the classical world by Canetti [Can01] and independently (under the name of reactive simulatability) by Pfitzmann and Waidner [PW01]. The model was extended to the quantum world by Ben-Or and Mayers [BOM04] and independently by Unruh [Unr04]. Here, we follow the simplified UC framework as presented by Unruh [Unr10], to which we refer for the description of the model, definitions, and theorems.

Via the composition theorem, this framework allows to rigorously prove results that are maximally useful for future work: our results can easily be embedded within a larger construction and at the same time we can use prior constructions without having to re-visit their security proofs. Another key result is Unruh's *quantum lifting theorem* [Unr10] establishing that, in the statistical case, classical-UC-secure protocols are also quantum-UC-secure.

Our main possibility result heavily relies on classical one-time programs, for which a (classical) UC-secure instantiation exists assuming one-time memories [GIS+10]. In particular, we require bounded reactive OTPs, which we construct by extending the results of Goyal *et al.* [GIS+10] (see Section 2.3).

Some notable variations of Unruh's terminology and definitions follow.

Out of the two protocol parties (the *sender* and the *receiver*), we consider security only in the case of the receiver being a corruption party. This is the meaning of, *e.g.* "π *statistically*

quantum-UC-emulates ρ in the case of a corrupted receiver".

In our scenario involving a corrupted receiver, we make use of a property related to the transitivity of UC-security [Unr10] (see Lemma 1); the proof is very similar to the original proof of transitivity.

Lemma 1. *Let π, σ and ρ be protocols with parties $\{\texttt{sender}, \texttt{receiver}\}$. Suppose that π statisically quantum-UC-emulates ρ and that ρ statistically quantum-UC-emulates σ in the case of a corrupted receiver. Then π statistically quantum-UC-emulates σ in the case of a corrupted receiver.*

Combining Lemma 1 with the quantum universal composition theorem [Unr10] and the quantum lifting theorem [Unr10] we get Corollary 1.1 below, which is essential in the proof of our main possibility theorem (Theorem 7).

Corollary 1.1. *Let π, and ρ be protocols with parties $\{\texttt{sender}, \texttt{receiver}\}$. Suppose π statistically classical-UC-emulates $\mathcal{F}$. Suppose that $\rho^{\mathcal{F}}$ statistically quantum-UC-emulates $\mathcal{G}$ in the case of a corrupted receiver. Then ρ^{π} statistically quantum-UC-emulates $\mathcal{G}$ in the case of a corrupted receiver.*

2.2 Ideal functionalities

We now describe the relevant ideal functionalities. All functionalities involve two parties, the *sender* and the *receiver*. An ideal functionality may exist in multiple instances and involves various parties. Formally, instances are denoted by *session identifiers* and each instance involves labelled parties. For the sake of simplicity, we have omitted these identifiers as they should be implicitly clear from the context.

The ideal functionality $\mathcal{F}^{\mathrm{OTM}}$ for a one-time memory (OTM) is a two-step process modelled after oblivious transfer. We sometimes refer to this functionality $\mathcal{F}^{\mathrm{OTM}}$ as an *OTM token.*

Functionality 1 Ideal Functionality $\mathcal{F}^{\mathrm{OTM}}$.

1. **Create:** Upon input (s_0, s_1) from the sender with $s_0, s_1 \in \{0,1\}$, send `create` to the receiver and store (s_0, s_1).
2. **Execute:** Upon input $c \in \{0,1\}$ from the receiver, send $s := s_c$ to the receiver. Delete any trace of this instance.

Next we describe the ideal functionalities $\mathcal{F}_f^{\mathrm{OTP}}$ and $\mathcal{F}_\Phi^{\mathrm{OTP}}$ of a one-time program for a classical function f and a quantum channel Φ, respectively. Note that the map that is computed (f or Φ) is a public parameter of the functionality and it takes an input from the sender and an input from the receiver. We thus view these ideal functionalities as having the property of hiding the sender's *input* only. If the intention is to to hide the map m itself—as in the intuitive notion of one-time programs—then we can consider a universal map U that takes as part of its sender's input a program register representing m (see [BFGH10, NC97, dSR07]).

Functionality 2 Ideal functionality $\mathcal{F}_f^{\mathrm{OTP}}$ for a classical function $f : \{0,1\}^{n+m} \to \{0,1\}^k$.

1. **Create:** Upon input $a \in \{0,1\}^n$ from the sender, send `create` to the receiver and store a.
2. **Execute:** Upon input $b \in \{0,1\}^m$ from the receiver, send $f(a,b)$ to the receiver. Delete any trace of this instance.

Functionality 3 Ideal functionality $\mathcal{F}_{\Phi}^{\mathrm{OTP}}$ for a quantum channel $\Phi : (\mathsf{A}, \mathsf{B}) \to \mathsf{C}$.

1. **Create:** Upon input register A from the sender, send `create` to the receiver and store the contents of register A.
2. **Execute:** Upon input register B from the receiver, evaluate Φ on registers A, B and send the contents of the output register C to the receiver. Delete any trace of this instance.

It is clear from the description of these ideal functionalities that they may be called only a single time. However, we may sometimes emphasize this in expressions such as "one-shot access to an ideal functionality computing f".

Functionalities 1–3 are *sender-oblivious* since they take an input from the sender and an input from the receiver and deliver the result of the functionality to the receiver (but not the sender). Moreover, they are *non-reactive* since they interact with the sender and the receiver in a single round. *Reactive* functionalities are more general, potentially having several rounds of inputs and outputs and maintaining state between rounds. In Section 2.3 we consider an ideal functionality for *bounded reactive classical one-time programs*; the ideal functionality for bounded-reactive OTPs is specified in Appendix A.

2.3 Classical one-time programs

Our construction relies heavily on classical OTPs, the construction of which is given by Goyal *et al.* [GIS+10]:

Theorem 2. *Let f be a non-reactive, sender-oblivious, polynomial-time computable classical two-party functionality. Then there exists an efficient, non-interactive protocol which statistically classical-UC-emulates $\mathcal{F}_f^{\mathrm{OTP}}$ in the $\mathcal{F}^{\mathrm{OTM}}$-hybrid model.*

In Appendix A, we use straightforward techniques to extend this result to sender-oblivious, polynomial-time computable, *bounded reactive* classical two-party functionalities. The main result on reactive OTPs, as used in our construction in Section 4, is:

Corollary 2.1. *There exists a non-interactive protocol σ that statistically classical-UC-emulates $\mathcal{F}_{g_1,\ldots g_\ell}^{\text{BR-OTP}}$ in the $\mathcal{F}^{\mathrm{OTM}}$-hybrid model.*

3 Impossibility of non-trivial OTPs in the plain model

We now consider whether classical functions or quantum channels admit one-time programs in the plain quantum model. We will see that it is precisely maps that are *unlockable*—meaning there is a *key*[1] input to the map that unlocks enough information to fully simulate the map—that have one-time programs in the plain model, and that these one-time programs are in a sense trivial. For quantum channels, we will have two versions of unlockable, the difference being whether the key that unlocks the channel is a state (*strongly unlockable*) or a channel that transforms a given input (*weakly unlockable*).

Our *possibility* result shows that every strongly unlockable channel admits a trivial one-time program in the plain quantum model, and in fact that this protocol is UC-secure. Our *impossibility* result shows that every channel that is not weakly unlockable does not admit a one-time program

[1]Note we use "key" not in the cryptographic sense of a secret key, but in the metaphorical sense of something that unlocks a lock.

in the plain quantum model; our impossibility result holds even if we relax to an approximate case or allow computational assumptions.[2] As we will see, it is easy to establish that the weak and strong unlockable notions are equivalent for classical functions, but whether they are equivalent for quantum channels is an open question, which we note appears to be an interesting and deep question related to invertible subspaces of a channel.

3.1 One-time programs in the plain quantum model

Here, we formalize some concepts relating to one-time programs in the plain quantum model.

A protocol for a quantum one-time program in the plain model consists of a single quantum message, the *program register* P, from the sender to receiver. Thus, the actions of the honest sender in such a protocol are completely characterized by an *encoding channel* $\mathrm{enc} : \mathsf{A} \to \mathsf{P}$ that the sender applies to her portion of the joint input so as to prepare a program register for the receiver. In particular, for any joint state ρ of the input registers (A, B) the joint state of the registers (P, B) in the receiver's possession after the program register has been received is given by $(\mathrm{enc} \otimes \mathbb{1}_{\mathsf{B}})(\rho)$. From this state we would like the receiver to be able to recover $\Phi(\rho)$. That is, there should exist a *decoding channel* $\mathrm{dec} : (\mathsf{P}, \mathsf{B}) \to \mathsf{C}$ for the honest receiver such that the channels $\mathrm{dec} \circ \mathrm{enc}$ and Φ are indistinguishable. A comparison of the ideal and real models for one-time programs is given in Figure 1.

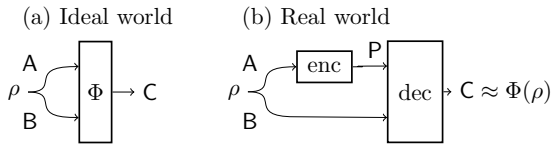


Figure 1: (a) In the ideal world, the receiver obtains the output of the ideal functionality for Φ on arbitrary input registers (A, B). (b) In the real world, encoding and decoding maps implement the functionality, namely $\mathrm{dec} \circ \mathrm{enc} \approx \Phi$.

By the completeness of the dummy-adversary [Unr10], it is sufficient, in order to establish UC-security, to consider only the case of the dummy-adversary who forwards the program register, P, to the environment. Thus, UC-security is established by exhibiting a simulator that can re-create a state that is indistinguishable from the joint state $(\mathrm{enc} \otimes \mathbb{1}_{\mathsf{B}})(\rho)$ of registers (P, B), using only the ideal functionality; recall indistinguishability is from the perspective of the environment, and could be perfect, statistical, or computational as appropriate. The corresponding channels are depicted in Figure 2. Here, the *simulator* $(\mathrm{sim}_1, \mathrm{sim}_2)$ consists of channels $\mathrm{sim}_1 : \mathsf{B} \to (\mathsf{B}', \mathsf{M})$ and $\mathrm{sim}_2 : (\mathsf{C}, \mathsf{M}) \to (\mathsf{P}, \mathsf{B})$, where M is a private memory register for the simulator, security holds if the channels $\mathrm{sim}_2 \circ \Phi \circ \mathrm{sim}_1$ and $\mathrm{enc} \otimes \mathbb{1}_{\mathsf{B}}$ are indistinguishable.

3.2 Trivial one-time programs for unlockable channels

Intuitively, a one-time program for a channel means that no receiver can learn more than he could given one-shot access to an ideal functionality. However, there are certain channels where one-shot access to the ideal functionality is enough to fully simulate the map for a fixed choice of the sender's

[2] Although our impossibility result is stated in the UC framework, the impossibility is not an artifact of the high level of security required of UC, but seems inherent in the notion of OTPs, and the impossibility argument applied for any relaxation we attempted.

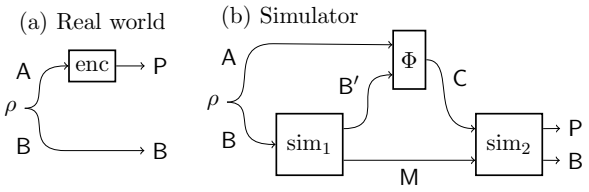


Figure 2: (a) The sender prepares the program register P by applying enc to A. The sender cannot touch B. (b) A simulator $(\mathrm{sim}_1, \mathrm{sim}_2)$ should be able to re-create an indistinguishable state of (P, B) using only the ideal functionality Φ.

input. Such channels—which we call *unlockable*—effectively have trivial one-time programs, as we will see in this section.

Definition 3. *A channel* $\Phi : (\mathsf{A}, \mathsf{B}) \to \mathsf{C}$ *is* strongly unlockable *if there exists a register* K*, a* key state ξ_0 *of* (B, K) *and a* recovery algorithm *(*i.e.*, channel)* $\mathcal{A} : (\mathsf{C}, \mathsf{K}, \mathsf{B}) \to \mathsf{C}$ *with the property that* $\mathcal{A} \circ \Phi_0 \approx \Phi$*, where the channel* Φ_0 *is specified by*

$$\Phi_0 : \mathsf{A} \to (\mathsf{C}, \mathsf{K}) : A \mapsto (\Phi \otimes \mathbb{1}_{\mathsf{K}})(A \otimes \xi_0)\,.$$

A channel $\Phi : (\mathsf{A}, \mathsf{B}) \to \mathsf{C}$ *is* weakly unlockable *if there exists a register* K *and a* key channel $\Xi_0 : \mathsf{B} \to (\mathsf{B}, \mathsf{K})$ *such that the channel* $\Phi \circ \Xi_0$ *has the following property: for every choice of registers* E *and channels* $\Psi : \mathsf{B} \to (\mathsf{B}, \mathsf{E})$ *for the receiver there exists a* recovery algorithm *(*i.e.*, channel)* $\mathcal{A}_\Psi : (\mathsf{C}, \mathsf{K}) \to (\mathsf{C}, \mathsf{E})$ *such that:*

$$\mathcal{A}_\Psi \circ \Phi \circ \Xi_0 \approx \Phi \circ \Psi\,.$$

Here, $\approx$ *can denote perfect, statistical, or computational indistinguishability; in all cases, channels* Φ_0*,* $\mathcal{A}$*,* Ξ_0*, and* $\mathcal{A}_\Psi$ *must be efficient.* □

See Figure 3 for diagrams representing strongly and weakly unlockable channels.

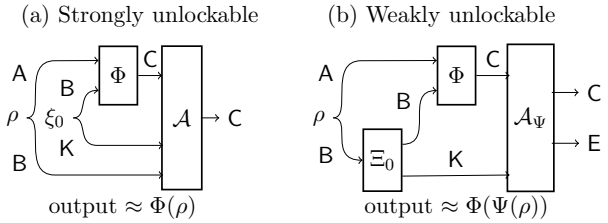


Figure 3: (a) For a strongly unlockable channel Φ, there exists a key state ξ_0 and a recovery algorithm $\mathcal{A}$ that allows computation of $\Phi(\rho)$ for any ρ. (b) For a weakly unlockable channel Φ, there exists a key channel Ξ_0 such that for any channel Ψ there exists a recovery algorithm $\mathcal{A}_\Psi$ that allows computation of $\Phi(\Psi(\rho))$ for any ρ.

It is easy to see that every strongly unlockable channel is also weakly unlockable: if ξ_0 is the key state for Φ, then the key channel Ξ_0 generates ξ_0, sends the B register of ξ_0 to ideal functionality Φ and the K register of ξ_0 and the B register of ρ to $\mathcal{A}_\Psi = \mathcal{A} \circ \Psi$.

When the channel Φ is an entirely classical mapping, the definitions of strongly unlockable and weakly unlockable are equivalent. A simplification for the classical case is as follows (we restrict to

the perfect case for clarity). A classical function $f : A \times B \to C$ is *unlockable* if there exists a *key input* $b_0 \in B$ and a *recovery algorithm* $\mathcal{A} : C \times B \to C$ such that, for all $a \in A$ and $b \in B$, we have that $f(a, b) = \mathcal{A}(f(a, b_0), b)$. Intuitively, for an unlockable function, there exists an algorithm that can compute all values of $f(a, \cdot)$ given a one-time program for $f(a, \cdot)$, but this is okay, because a simulator, given one-shot oracle access to $f(a, \cdot)$, can also compute $f(a, b)$ for all b. This function is "learnable" in one shot, and so a simulator can do everything any algorithm can.

Simple examples of strongly unlockable channels include all unitary channels of the form $\Phi : X \mapsto UXU^*$ for some unitary U and all constant channels of the form $\Phi : X \mapsto \mathrm{Tr}(X)\sigma$ for some fixed state σ. Simple examples of unlockable functions include permutations.

We can now see that strongly unlockable channels have one-time programs; in fact, trivial one-time programs.

Theorem 4. *Let $\Phi : (\mathsf{A}, \mathsf{B}) \to \mathsf{C}$ be a non-reactive, sender-oblivious polynomial-time quantum computable two-party functionality. Then if Φ is strongly unlockable, there exists an efficient, quantum non-interactive protocol which quantum-UC-emulates $\mathcal{F}_\Phi^{\mathrm{OTP}}$ in the plain quantum model.*

This holds in the perfect, statistical and computational cases.

Proof. The protocol is simple.

1. **Create:** The sender prepares the program register $\mathsf{P} = (\mathsf{C}, \mathsf{K})$ by preparing registers $(\mathsf{B}', \mathsf{K})$ in the key state ξ_0 and applying Φ to $(\mathsf{A}, \mathsf{B}')$. In other words, the sender's encoding channel enc is given by $\mathrm{enc} = \Phi_0$, where Φ_0 is as in the definition of strongly unlockable.
2. **Execute:** Because Φ is strongly unlockable, the receiver can recover the action of $\Phi = \mathcal{A} \circ \Phi_0$ simply by applying the recovery algorithm $\mathcal{A}$ to (P, B). In other words, the receiver's decoding channel dec is given by $\mathrm{dec} = \mathcal{A}$.

Clearly, because Φ is strongly unlockable, the output of the honest receiver in the real model is indistinguishable from the output of the ideal model.

According to the discussion in Section 3.1, in order to show that this protocol is secure, it suffices to exhibit a simulator that can emulate the channel $\mathrm{enc} \otimes \mathbb{1}_\mathsf{B} = \Phi_0 \otimes \mathbb{1}_\mathsf{B}$ using only the ideal functionality. But this is easy: the simulator can emulate $\Phi_0 \otimes \mathbb{1}_\mathsf{B}$ simply by preparing registers $(\mathsf{B}', \mathsf{K})$ in the key state ξ_0 and using the ideal functionality. Formally, the simulator $(\mathrm{sim}_1, \mathrm{sim}_2)$ is specified by $\mathrm{sim}_1 : \mathsf{B} \to (\mathsf{B}', \mathsf{K}, \mathsf{B}) : X \mapsto \xi_0 \otimes Y$ and $\mathrm{sim}_2 = I$, so that

$$\mathrm{sim}_2 \circ \Phi \circ \mathrm{sim}_1 : X \otimes Y \mapsto (\Phi \otimes \mathbb{1}_\mathsf{K})(X \otimes \xi_0) \otimes Y = \Phi_0(X) \otimes Y$$

as desired. (See Figure 2.) □

Theorem 4 is in the quantum model; it is not hard to see by its proof that if Φ is in fact a classical channel, then the resulting protocol π is a purely classical protocol.

3.3 Impossibility of one-time programs for arbitrary channels

Having seen that, in the plain model, one-time programs do exist for strongly unlockable channels, we now see that they do not exist for weakly unlockable channels.

Theorem 5. *Suppose $\Phi : (\mathsf{A}, \mathsf{B}) \to \mathsf{C}$ is a non-reactive, sender-oblivious polynomial-time quantum computable two-party functionality, and suppose that Φ admits an efficient, non-interactive quantum protocol which quantum-UC-emulates $\mathcal{F}_\Phi^{\mathrm{OTP}}$ in the plain model. Then Φ is weakly unlockable.*

This holds in the perfect, statistical and computational cases.

The intuition of the proof is as follows. If a channel has a one-time program, then for any adversary there exists a simulator that can match the behaviour of the adversary. In particular, there must be a simulator that matches the behaviour of the dummy adversary that just outputs the program state: thus, there must be an algorithm that can reconstruct the program state given the output of the channel, thus allowing computation for any output, meeting the definition of a weakly unlockable channel.

Proof. Suppose Φ is as in the theorem statement. By the discussion in Section 3.1, there exists an encoding channel $\mathsf{enc} : \mathsf{A} \to \mathsf{P}$ for the sender and a decoding channel $\mathsf{dec} : (\mathsf{P}, \mathsf{B}) \to \mathsf{C}$ for the receiver such that $\mathsf{dec} \circ \mathsf{enc}$ is indistinguishable from Φ. Moreover, there exists a simulator $(\mathrm{sim}_1, \mathrm{sim}_2)$ with $\mathrm{sim}_2 \circ \Phi \circ \mathrm{sim}_1$ being indistinguishable from $\mathsf{enc} \otimes \mathbb{1}_{\mathsf{B}}$.

We claim that Φ is weakly unlockable with key channel $\Xi_0 = \mathrm{sim}_1$ and recovery algorithm given by $\mathcal{A}_\Psi = \mathsf{dec} \circ \Psi \circ \mathrm{sim}_2$ for any choice of Ψ. To this end fix a choice of $\Psi : \mathsf{B} \to (\mathsf{B}, \mathsf{E})$. The channels Ψ and enc commute, as they act on different input registers. Thus using $\approx$ to indicate computational, statistical, or perfect indistinguishability, depending on the scenario,

$$\Phi \circ \Psi \approx \mathsf{dec} \circ \mathsf{enc} \circ \Psi = \mathsf{dec} \circ \Psi \circ \mathsf{enc} \approx \underbrace{\mathsf{dec} \circ \Psi \circ \mathrm{sim}_2}_{\mathcal{A}_\Psi} \circ \Phi \circ \underbrace{\mathrm{sim}_1}_{\Xi_0} . \qquad \square$$

An alternate intuition for the impossibility result for classical functions can be found by considering rewinding. Any correct one-time program state ρ_x for a classical function $f(x, \cdot)$ must result in the receiver obtaining an output state $\rho_{x,y}$ that is (almost) orthogonal in the basis in which the receiver measures it, because the measurement of $\rho_{x,y}$ results in $f(x, y)$ with (almost) certainty. As a result, measurement does not disturb the state (much), so the receiver can reverse the computation to obtain (almost) the program state again, and then rerun the computation to obtain $f(x, y')$ for a different y'. It is possible to give a proof for impossibility of OTPs for classical functions in the plain quantum model using this rewinding argument. Impossibility for classical functions also follows as a special case of the impossibility shown in [BCS12].

3.4 A conjecture on unlockable channels

As noted earlier, every strongly unlockable channel is also weakly unlockable. We conjecture that the converse also holds: every weakly unlockable channel is also strongly unlockable. Though we do not yet have a formal proof of this conjecture for arbitrary Φ, we can nonetheless provide a high-level outline of what such a proof might look like.

Conjecture 6. *Every channel $\Phi : (\mathsf{A}, \mathsf{B}) \to \mathsf{C}$ that is weakly unlockable is also strongly unlockable.*

Proof outline. Let σ be any mixed state of a register $\mathsf{B}' \equiv \mathsf{B}$ and consider the channel

$$\Psi_{\mathrm{dummy}} : \mathsf{B} \to (\mathsf{B}', \mathsf{B}) : B \mapsto \sigma \otimes B$$

so that

$$\Phi \circ \Psi_{\mathrm{dummy}} : (\mathsf{A}, \mathsf{B}) \to (\mathsf{C}, \mathsf{B}) : A \otimes B \mapsto \Phi(A \otimes \sigma) \otimes B.$$

That is, the channel Ψ_{dummy} swaps out the receiver's input register B with a dummy register B' in state σ before Φ is applied. By definition, the channel $\Phi \circ \Psi_{\mathrm{dummy}}$ does not touch B.

Let $\Xi_0, \mathcal{A}_\Psi$ witness the weak unlockability of Φ. For the choice $\Psi = \Psi_{\text{dummy}}$ it holds that the channel

$$\mathcal{A}_{\Psi_{\text{dummy}}} \circ \Phi \circ \Xi_0 = \Phi \circ \Psi_{\text{dummy}}$$

also must not touch B. Hence it must be that the channel $\Phi \circ \Xi_0$ is invertible on B.

By enlarging the key register K as needed, we may assume without loss of generality that Ξ_0 is implemented by an isometry S. Thus, in order to be invertible on B it must be that for each pure state $|\psi\rangle$ of B the state $S|\psi\rangle$ of (B, K) can be recovered after Φ acts on B. In other words, Φ must be invertible on the subspace $\mathcal{B}_{\text{inv}}$ of the state space of register B containing the image of $\text{Tr}_{\mathsf{K}}(S|\psi\rangle\langle\psi|S^*)$ for every choice of $|\psi\rangle$.

Before putting B through Φ an alternate key channel Ξ_0' could coherently swap out the subspace $\mathcal{B}_{\text{inv}}$ with some portion of the key register K in any fixed state ξ. Then, after receiving C from Φ, the alternate recovery algorithm $\mathcal{A}'_{\Psi_{\text{dummy}}}$ could perform the inversion operation to recover ξ, swap this state back into the key register, and then perform the inverse of the inversion operation on $S|\psi\rangle$ so as to apply Φ to this state. These modifications yield the desired simulator. □

For the proof of the conjecture to go through, we need the invertibility of Φ on the aforementioned substance $\mathcal{B}_{\text{inv}}$ to hold. It appears, then, that we have stumbled upon an interesting and deep question relating to the invertible subspaces of a channel, akin to the "decoherence-free" subspaces studied in the literature on quantum error correction.

4 Constructing quantum OTPs from OTMs

We now state our main possibility theorem which establishes non-interactive unconditionally secure quantum computation using OTM tokens.

Theorem 7. *Let Φ be non-reactive, sender-oblivious polynomial-time quantum computable two-party functionality. Then there exists an efficient, quantum non-interactive protocol which statistically quantum-UC-emulates $\mathcal{F}_\Phi^{\text{OTP}}$ in the case of a corrupt receiver, in the $\mathcal{F}^{\text{OTM}}$-hybrid model.*

The proof of Theorem 7 follows directly from Theorem 8 below, together with Corollary 2.1, the quantum lifting theorem as well as Lemma 1.

Theorem 8. *Let Φ be a non-reactive, sender-oblivious polynomial-time quantum computable two-party functionality. Then there exists an efficient, statistically quantum-UC-secure non-interactive protocol which realizes $\mathcal{F}_\Phi^{\text{OTP}}$ in the case of a corrupt receiver, in the $\mathcal{F}^{\text{BR-OTP}}$-hybrid model.*

The proof of Theorem 8 is presented in the following sections, which we briefly highlight here; a detailed outline follows in the next section.

1. Section 5 presents our new *trap authentication scheme*, a type of quantum authentication code. We show how perform a universal set of quantum gates (X, Y, Z, CNOT, i-shift and $\pi/8$ phases, and H) on authenticated data without knowing the authentication key.
2. Section 6 presents our protocol for quantum one-time programs and the proof its security. Since computation on authenticated data requires updates to be performed that are dependent on the authentication key, our protocol uses a reactive classical one-time program (based on one-time memories) to allow the receiver to non-interactively implement the required operations to correctly compute on the sender's authenticated data.

The following sections 4.1–4.5 provide an overview of the proof and related techniques.

4.1 Quantum authentication codes

A quantum authentication scheme consists of procedures for encoding and decoding quantum information with a secret classical key k such that an adversary with no knowledge of k who tampers with encoded data will be detected with high probability. Quantum authentication codes were first introduced by Barnum, Crépeau, Gottesman, Smith and Tapp [BCG+02].

Some of the known quantum authentication schemes have the following general form. Quantum information is encoded according to some quantum error detecting code E chosen uniformly at random from a special family $\mathscr{E}$ of codes. The encoded quantum data is then encrypted according to the quantum one-time pad, meaning that a uniformly random Pauli operation P is applied to the data encoded under E. The secret classical key for schemes of this form is the pair (E, P) describing the choice of code E and encryption Pauli P. Authenticated quantum data is later verified by decrypting according to P and then decoding according to E. Verification passes only if the error syndrome for E indicates no errors. **Terminology:** In this paper authentication schemes of this form are called *encode-encrypt schemes*.

This construction is desirable due to the remarkable property (known as the *Pauli twirl* [DCEL09]) that the Pauli encryption serves to render *any attack* on the scheme equivalent to a *probabilistic Pauli attack* on data encoded with a random code $E \in \mathscr{E}$. Thus, to establish a secure authentication scheme one need only construct a family $\mathscr{E}$ of quantum error detecting codes that detect Pauli attacks with high probability over the choice of $E \in \mathscr{E}$.

Our new *trap authentication scheme* falls in this family of codes. The family of codes $\mathscr{E}$ is based on any quantum error detecting code C with distance d that encodes a single qubit into n qubits. Authentication consists of first encoding a qubit under C and then appending n qubits set to $|0\rangle$ and n qubits set to $|+\rangle$. A random permutation (indexed by a classical key) is then applied. The first use of this code was implicit in the Shor–Preskill security proof for quantum key distribution [SP00] (see also [BFK09]).

4.2 Computing on authenticated data

At a high level, our main protocol uses quantum authentication codes in order to protect the data from any tampering by the receiver during the computation. An authentication code is insufficient, however, because we want to implement a channel on this authenticated data, as specified by a quantum circuit. For this, we use techniques for quantum computing on authenticated data, as first established for the *signed polynomial code* [BOCG+06] (see also [ABOE10]), and recently (and independently of our work), for the Clifford authentication code [DNS12]. More specifically, computing on authenticated data allows acting on the encoded registers in order to implement a known gate, *but without knowledge of the key*. Normally, any non-identity operation would invalidate the authenticated state, but our encoded operations, together with a *key update* operation, effectively *forces* the application of the desired gate, as otherwise the state would fail verification under the updated key.

Encoded gates are executed in a manner similar to encoded gates in fault-tolerant quantum computation: some gates (such as Pauli gates or the controlled-not gate) are transversal, while other (such as the $\pi/8$ gate) require both an auxiliary register and classical interaction with an entity who knows the encoding keys. This classical interaction makes our quantum one-time program "interactive", but only at a classical level. Thus, by extending classical one-time programs to *reactive* functionalities, we manage to encapsulate this interaction into a classical one-time program.

Comparison with other methods of computing on authenticated data. Although methods for computing on authenticated data were developed prior to our work, we believe that the simplicity of our trap scheme is an advantage. For example, whereas the trap scheme is defined for qubits, the signed polynomial scheme of Ben-Or *et al.* [BOCG+06] acts on d-dimensional qudits with d dictating the security of the scheme. By necessity, the universal gate set U_d for the polynomial scheme is different for each d. In order to use the polynomial scheme for computation on qubits, one must first embed the desired qu*bit* computation into a qu*dit* computation. (For example, a naive approach is to simply embed one qubit into each qudit and use only the first two dimensions.) However this embedding is chosen, one must demonstrate that gates in the original qubit computation can be implemented efficiently using gates from U_d. Normally, an efficient transition between universal gate sets is implied by the Solovay–Kitaev algorithm, but this techniques scales poorly with d and so cannot be used for this purpose for large d. (See Dawson and Nielsen [DN06] and the references therein.) Although we fully expect such an embedding to admit an efficient implementation, it appears that the issue has not been addressed in the literature.

Compared with the gate implementations in the recent Clifford scheme of Dupuis *et al.* [DNS12], our trap scheme is less complex and requires less communication between the sender and the receiver (the Clifford scheme requires communication for *all* circuit elements).

4.3 Gate teleportation

The main outline of our protocol is now becoming clearer: the receiver executes the encoded circuit, using techniques for computing on authenticated data. But how does the receiver get the authenticated version of her data in the first place? And how does the receiver get the decoded output? We resolve this by using encoding and decoding *gadgets* that are inspired by the gate teleportation technique of Gottesman and Chuang [GC99]. In this technique, a quantum register undergoes a transformation by a quantum circuit via its teleportation through a special entangled state. In our case, encoding is performed by teleporting the input qubit through an EPR pair, half of which has itself undergone the encoding operation. By revealing the classical result of the teleportation, the authentication key for the output of this process is determined. Decoding is similar. The encoding and decoding gadgets are prepared by the sender as part of the quantum one-time program.

4.4 Overview of protocol

Given the tools and techniques described above, the structure of our protocol is as follows (although we suggest the use of the trap authentication scheme, the protocol and proof is applicable to any encode-encrypt quantum authentication scheme that admits computing on authenticated data).

1. The sender gives the receiver an authenticated version of the sender's input, together with auxiliary states required for evaluating the target circuit. The sender also prepares encoding and decoding gadgets.
2. The sender gives the receiver a bounded reactive classical one-time program that emulates the classical interaction that would occur when using the encoding and decoding gadgets, as well as for computing on authenticated data.
3. The receiver uses the encoding gadget to encode his input; he then performs the target circuit on the authenticated data by performing encoded gates. Finally, he decodes the output using the decoding gadget. All classical interaction is done via the classical one-time program.

As a proof technique, we specify that the target circuit be given as a controlled-unitary, with the control set to 1, for a reason described in the simulation sketch below.

4.5 Overview of simulator for proof

In order to prove UC security, we must establish that every real-world adversary has an ideal-world simulator. This is done via a rigorous mathematical analysis of any arbitrary attack; the exposition of our proof is aided by a *table representation* we have developed for pure quantum states (see Section C.2). The simulator prepares a quantum one-time program as in Section 4.4, with the following modifications:

1. The encoding gadget is split into two halves. The first half is a simple EPR pair, used to *extract* the input from the adversary; the second half is an encoding gadget, used to *insert* the output of the ideal functionality into the computation.
2. The control-bit for the controlled-unitary is 0.
3. The encoded input is a dummy encoded input.

The simulator then executes the adversary on this "quantum one-time program". After the use of the encoding gadget, the receiver's input is determined and this is used as input into the ideal functionality. The output of the ideal functionality is then returned into the computation via the encoding gadget. Because the control bit is set to 0, the computation actually performs the identity. When the adversary is honest, the output will therefore be correct. For any behaviour of the adversary, our analysis shows that the ideal and real worlds are indistinguishable. The simulator thus indistinguishably emulates the real-world behaviour of any adversary with just a single call to the ideal functionality, which establishes UC security.

5 The trap authentication scheme

In this section we introduce the trap authentication scheme, which is an example of an encrypt-encode scheme as described in Section 4.1. We show how to implement gates from a universal set on data authenticated under this scheme (including measurement in the standard basis), from which it follows that the trap scheme admits QCAD.

5.1 Definitions and notation

In this paper we assume that a quantum error correcting code is specified by a unitary operation E that can be implemented by a circuit consisting entirely of Clifford gates. A data register D is encoded under code E by preparing two syndrome registers (X, Z) in the $|0\rangle$ state and applying E to $(\mathsf{D}, \mathsf{X}, \mathsf{Z})$. Data is decoded by applying the inverse circuit E^* and measuring the syndrome registers (X, Z) in the computational basis. Any non-zero syndrome measurement result indicates an error (or that cheating has been detected, depending on the context). These assumptions are met, for example, by every stabilizer code. In order to minimize the number of symbols for distinct quantum registers we adopt the convention that the tilde $\tilde{}$ denotes an *encoded register*. For example, the encoded register $\tilde{\mathsf{D}}$ consists of a data portion D plus two syndrome registers X, Z and can be viewed as a triple $\tilde{\mathsf{D}} = (\mathsf{D}, \mathsf{X}, \mathsf{Z})$.

Let E be a code for some data register D and let Q be any Pauli acting on $\tilde{\mathsf{D}}$. As E is a Clifford circuit there must be a Pauli Q_E acting on $\tilde{\mathsf{D}}$ with $QE = EQ_E$.

Definition 9. *A family $\mathscr{E}$ is said to be* ϵ-secure against Pauli attacks *if for each fixed choice of Pauli Q acting on $\tilde{\mathsf{D}}$ it holds that the probability (taken over a uniformly random choice of code $E \in \mathscr{E}$) that Q_E acts nontrivially on logical data and yet has no error syndrome is at most ϵ.* □

Formally defining the security of an authentication scheme is tricky work. (See Barnum *et al.* [BCG+02] for a discussion of definitional issues.) Fortunately, the task becomes much easier if we restrict attention to encode-encrypt schemes described in Section 4.1. This happy state of affairs is a consequence of the fact that an arbitrary attack on data authenticated under an encode-encrypt scheme is equivalent to a probabilistic mixture of Pauli attacks. (See below for futher discussion.) Thus, an encode-encrypt scheme is ϵ-secure against against arbitrary attacks if and only if the underlying family $\mathscr{E}$ of codes is ϵ-secure against Pauli attacks.

5.2 Properties of every encode-encrypt authentication scheme

Before introducing the trap scheme we review some facts about encrypt-encode authentication schemes as described in Section 4.1. These facts were largely known to the community prior to the present work and hence a full discussion is relegated to Appendix B. Our purpose in this paper is to collect these facts in one convenient place and to present them in a way that is amenable to a discussion of QCAD. In Appendix B we formalize and prove the following facts:

1. Any family $\mathscr{E}$ of codes that is ϵ-secure against Pauli attacks immediately induces an ϵ-secure encode-encrypt scheme via the construction described in Section 4.1.
2. If the codes in $\mathscr{E}$ are CSS codes then measurement of logical data in the computational basis can be implemented by bitwise measurement of authenticated data followed by a classical decoding process in order to determine the measurement result.
3. The measure-then-decode procedure of property 2 is equivalent to a decode-then-measure procedure in which the register $\tilde{\mathsf{D}}$ is first de-authenticated and then the logical register D is measured in the computational basis. In this procedure, the X-syndrome register is also measured in the computational basis so as to check for errors, but the Z-syndrome register is simply discarded without any verification.
4. For any encode-encrypt scheme thus constructed, the measure-then-decode procedure (or equivalently, the decode-then-measure procedure) of property 2 is also ϵ-secure against arbitrary attacks.
5. In any encode-encrypt scheme the code key E can be re-used to authenticate multiple distinct data registers, provided that each new register gets its own fresh Pauli key P.

The proof of property 5 is easy enough that we can give it immediately. As noted in Section 4.1 (and established in the proof of property 1 in Appendix B), the Pauli encryption serves to render any attack on an encode-encrypt scheme equivalent to a probabilistic mixture of Pauli attacks. By definition a Pauli attack is a product attack on each physical qubit in the authenticated registers, so security against attacks on one register implies security against attacks on all registers. This observation was originally made in the analysis of the polynomial authentication scheme [BOCG+06]. (See Ref. [ABOE10] for a slightly more detailed discussion.)

5.3 Trap codes yield a secure authentication scheme

In this section we describe a method by which any code E with distance d can be used to construct a family $\mathscr{E}$ of codes that is $(2/3)^{d/2}$-secure against Pauli attacks. Codes of this form shall be called *trap codes*. It follows immediately from the discussion of Section 5.2 that trap codes yield a secure

authentication scheme via the encode-encrypt construction of Section 4.1. This authentication scheme shall be called the *trap scheme*.

Furthermore, if the underlying code E is a CSS code then so is every member of the associated family $\mathscr{E}$ of trap codes. In this case it follows from the discussion of Section 5.2 that measurement of logical data in the computational basis can be implemented securely by bitwise measurement of physical data plus classical decoding.

For convenience we restrict attention to codes that encode one logical qubit into n physical qubits. (These are called $[[n, 1, d]]$-codes.) Given such a code, each member of the associated family $\mathscr{E}$ of trap codes is a $[[3n, 1, d]]$ code that is uniquely specified by a permutation π of $3n$ elements. In particular, for each π we construct a Clifford encoding circuit E_π as follows.

1. Encode the data qubit D under E, producing an n-qubit system $(\mathsf{D}, \mathsf{X}, \mathsf{Z})$.
2. Introduce two new n-qubit syndrome registers X', Z' in states $|0\rangle^{\otimes n}, |+\rangle^{\otimes n}$, respectively.
3. Permute all $3n$ qubits of $(\mathsf{D}, \mathsf{X}, \mathsf{Z}, \mathsf{X}', \mathsf{Z}')$ according to π.

The X- and Z-syndrome registers for the code E_π are $(\mathsf{X}, \mathsf{X}')$ and $(\mathsf{Z}, \mathsf{Z}')$, respectively so that $\tilde{\mathsf{D}} = (\mathsf{D}, (\mathsf{X}, \mathsf{X}'), (\mathsf{Z}, \mathsf{Z}'))$. Security of this family against Pauli attacks is easy to prove.

Proposition 10 (Security of trap codes against Pauli attacks)**.** *The family $\mathscr{E}$ of trap codes based on a code of distance d is $(2/3)^{d/2}$-secure against Pauli attacks.*

Remark. The bound in Proposition 10 is quite weak and can probably be strengthened significantly by a tighter analysis. All that really matters is that the security parameter decreases exponentially in d.

Proof. Let Q be a $3n$-qubit Pauli. In order for Q to act nontrivially on logical data it must have weight $w \geq d$, owing to the fact that the underlying code E has distance d. In this case Q must distribute w non-identity qubit Pauli operations over the $3n$ qubits without triggering any of the traps. Let us bound the probability of such an event.

In order to have weight w the Pauli Q must specify either an X-Pauli on at least $w/2$ qubits or a Z-Pauli on at least $w/2$ qubits. We analyze only the first case; a similar analysis applies to the second case. If any of these qubits belong to the register X' then Q will be detected as an error. Thus, to avoid detection all $w/2$ of these qubits must not belong to X'—a sample-without-replacement event whose probability of success is bounded by the probability of a successful sample-with-replacement event. The probability of success in any one sample is at most $2/3$ and so the probability of $w/2$ successful samples with replacement is at most $(2/3)^{w/2}$. □

5.4 Performing Gates on the Trap Code

Authentication schemes that also allow for the implementation of a universal set of quantum gates on authenticated data without knowing the key hold great promise for a host of cryptographic applications. In this section, we exhibit a universal gate set together with implementations of each gate in that set for the trap scheme. As discussed in Section 4.2, these techniques are used in our quantum one-time programs.

Let us be more explicit about what it means to apply gates to authenticated quantum data without knowing the key. It is helpful to think of two parties: a trusted *verifier* who prepares authenticated data with secret classical key k and a malicious *attacker* who is to act upon the authenticated data without knowledge of k. The goal is to construct an authentication scheme with the property that for each gate G belonging to some universal set of gates there exists a

gadget circuit $\tilde{G}$ that the attacker can perform on authenticated data so as to implement a logical G. Furthermore, we require that the gadget $\tilde{G}$ be *independent of the choice of classical key* k so that it may be implemented by an attacker without knowledge of k.

Normally, any non-identity gadget $\tilde{G}$ would invalidate the authenticated state. We therefore require a scheme which allows the verifier to validate the state again simply by updating the classical key $k \mapsto k'$. Moreover, by updating the key in this way the verifier effectively *forces* the attacker to apply the desired gadget $\tilde{G}$ as otherwise the state would fail verification under the updated key k'.

Some gadgets are quite simple. For example, we shall soon see that the gadget for a logical controlled-NOT in the trap scheme is simply a bitwise controlled-NOT applied to the physical qubits in the authenticated registers. Other gadgets, however, are more complicated, owing partly to the fact that there is no quantum error detecting code that admits bitwise implementation of every gate in a universal gate set.

Following the example of the polynomial scheme of Ben-Or *et al.* [BOCG+06], we borrow from the study of fault-tolerant quantum computation to complete a universal gate set by means of so-called "magic states". A *magic state gadget* for a logical gate G is a circuit that takes an additional (authenticated) ancillary register as input, performs a measurement, and then performs a correction based on the result of that measurement. A magic state gadget for G works only when the ancillae are prepared in a special (authenticated) *magic state* tailored specifically for the gate G. (For example, Ben-Or *et al.* exhibited a magic state and associated gadget for the generalized Toffoli gate under the polynomial scheme [BOCG+06].) Thus, implementation of a universal gate set on authenticated data requires the ability to prepare authenticated magic states and the ability to measure authenticated qubits in the computational basis.

5.5 A universal gate set for the trap scheme

In Section 5.3 we stipulated that a trap scheme can be constructed from any underlying code E chosen from a large class of codes. We also noted that if E is a CSS code then so are its associated trap codes, from which it follows that measurement of logical data can be implemented by a simple bitwise measurement of authenticated data.

In addition to being a CSS code, our implementation of a universal gate set for the trap scheme requires an underlying code E for which a logical Hadamard gate H is implemented by bitwise H on each physical qubit. CSS codes with this property are sometimes called *self-dual CSS codes*. The seven-qubit Steane code is one example of a self-dual CSS code that suffices for this purpose. Specifically, it follows from Proposition 10 that if our trap scheme is to have security $(2/3)^{d/2}$ then it suffices to base the trap scheme upon the Steane code nested a sufficient number of levels so as to achieve distance d.

Our universal gate set consists of the following gates, listed in the order they are presented in the following subsections.

1. The standard single-qubit Pauli gates, denoted X, Y, Z.
2. The standard two-qubit controlled-NOT gate, denoted CNOT.
3. The single-qubit i-shift phase gate, denoted K and specified by $K : |a\rangle \mapsto i^a|a\rangle$ for $a \in \{0, 1\}$.
4. The single-qubit $\pi/8$-phase gate, denoted T and specified by $T : |a\rangle \mapsto e^{ai\pi/4}|a\rangle$ for $a \in \{0, 1\}$.
5. The standard single-qubit Hadamard gate, denoted H.

This gate set is redundant in the sense that only $\{\text{CNOT}, H, T\}$ are required to achieve universality. Indeed, $T^2 = K$ and $K^2 = Z$, so why bother listing these extra gates? The answer is that the

gadgets for the Pauli and controlled-NOT gates are very simple. By contrast, the gadget for K is a magic state gadget that requires CNOT and Y and the gadget for T is a magic state gadget that requires CNOT, X, and K. Thus, in order to get a T gate we must "bootstrap" our way up from Pauli gates, CNOT, and K. (If an application calls for only Clifford circuits on authenticated data then the T gate construction can be ignored, as $\{\text{CNOT}, H, K\}$ suffice to generate the Clifford circuits.)

5.5.1 Pauli gates

The gadgets for each Pauli gate X, Y, Z are empty. As with the polynomial scheme [BOCG+06], in order to implement a logical Pauli gate in the trap scheme the attacker does absolutely nothing to the authenticated register and the verifier simply updates the Pauli key according to the desired Pauli gate.

In particular, logical Paulis for the seven-qubit Steane code admit straightforward bitwise implementations. Thus, the verifier can implement a logical Pauli Q in the trap scheme by modifying the Pauli key according to $P \mapsto PQ^{\otimes 7}$ where $Q^{\otimes 7}$ is applied to registers $(\mathsf{D}, \mathsf{X}, \mathsf{Z})$, leaving the Pauli key for the trap registers X', Z' unchanged.

5.5.2 The controlled-NOT gate

The gadget for a controlled-NOT from logical qubit a to logical qubit b is a straightforward bitwise CNOT applied from each physical qubit of a to its corresponding physical qubit in b.

To see that this simple bitwise gadget implements logical CNOT recall that every CSS code (including the Steane code) admits a bitwise implementation of logical CNOT. Moreover, the CNOT applied bitwise to the trap registers acts trivially on those registers:

$$\begin{aligned} \text{CNOT} &: |0\rangle|0\rangle \mapsto |0\rangle|0\rangle \\ &: |+\rangle|+\rangle \mapsto |+\rangle|+\rangle \end{aligned}$$

Finally, observe that bitwise CNOT is invariant under permutation of the physical qubits *provided that both data blocks are subjected to the same permutation.* (See Section 5.6.1 for further discussion.)

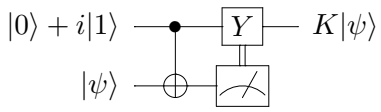
The Pauli key is updated according to the well-known effect of CNOT on Pauli operations. In particular, if the Pauli keys for the ith physical qubit from both data blocks are $X^p Z^q, X^r Z^s$, respectively, then the updated Pauli keys for these physical qubits are $X^p Z^{q+s}, X^{p+r} Z^s$.

5.5.3 The i-shift gate

Readers familiar with the Steane code know that a bitwise K gate applied to each physical qubit implements K^* on logical data. At first glance one might therefore hope that the K gate, like CNOT, admits a simple bitwise gadget under the trap scheme. Unfortunately, trap codes do not admit bitwise implementation of the K gate even if the underlying code does admit such an implementation. Bitwise implementation fails for trap codes because the trap qubits prepared in state $|+\rangle$ are mapped by K to $K|+\rangle = |0\rangle + i|1\rangle$. A trap qubit in this state is detected as a Z-error with probability $1/2$.

Instead we require a more complicated magic state gadget for K that uses only Pauli and CNOT gates together with measurement in the computational basis. Our gadget is a simple modification

of the well-known fault-tolerant construction for the $\pi/8$-gate of Boykin *et al.* [BMP+00]. The logical gadget for the K gate is depicted as follows.



(1)

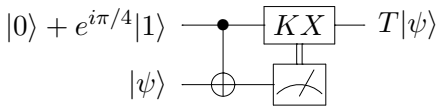
Here $|\psi\rangle$ denotes the arbitrary state of the data qubit; the magic state for this gadget is $|0\rangle + i|1\rangle$. The physical gadget to be implemented by the attacker on authenticated data is derived from the above logical gadget by replacing the input qubits with authenticated registers and by replacing the logical CNOT, Y, and measurement with their respective physical gadgets.

Once the authenticated magic state has been prepared, all the gates in this gadget except the measurement can be implemented by solely by the attacker. The verifier's knowledge of the secret key is required in order to decode the measurement result, which indicates whether a Y correction is needed.

Since Y is a Pauli gate and since Pauli gates require no action from the attacker, this gadget can be implemented with only one-way classical interaction from attacker to verifier. In particular, the attacker implements the measurement by bitwise measurement of physical data as described in Section 5.2. The verifier decodes the measurement result (and checks for tampering in the process) and then implements the Y correction (if it is needed) simply by updating the Pauli key for that qubit as described in Section 5.5.1.

5.5.4 The $\pi/8$-phase gate

The gadget for the T gate is a magic state gadget that is very similar to the magic state gadget for the K gate described Section 5.5.3 and consequently much of the discussion from that section applies here. The original fault-tolerant construction due to Boykin *et al.* [BMP+00] can be used verbatim as the logical gadget for the T gate in the trap scheme. Their construction is reproduced below.



(2)

The magic state for this gadget is $|0\rangle + e^{i\pi/4}|1\rangle$.

Whereas the gadget for K presented in Section 5.5.3 specified only a Pauli Y correction, the correction required in the T gadget is the Clifford gate KX. Two-way communication between verifier and attacker is required to implement this gadget because the verifier must inform the attacker as to whether to apply a K gate. Naturally, this K gate, if it is required, is achieved via the magic state gadget presented Section 5.5.3, which requires a separate magic state of its own.

Since K is not a Pauli gate subsequent computation on authenticated quantum data cannot proceed until the correction is applied (if it is needed). Thus, the verifier must decode the classical measurement result and reply immediately to the attacker with instructions as to whether a correction is required.

5.5.5 The Hadamard gate

Similar to the K gate, a logical H gate can be implemented under the Steane code by applying H bitwise to each physical qubit. One might therefore hope that the H gate admits a simple bitwise gadget under the trap scheme. Unfortunately, as with the K gate, trap codes do not admit bitwise implementation of H even if the underlying code does admit such an implementation. Bitwise implementation fails for trap codes because the $|0\rangle$ and $|+\rangle$ trap qubits are swapped by the action of bitwise H. Each trap qubit is thus in a state that is detected as an error with probability $1/2$.

As with the implementations of the K and T gates described previously we shall implement the Hadamard by a magic state gadget. Whereas the gadgets for K, T are compact and efficient, the simplest known magic state gadget for the Hadamard gate is the teleport-through-Hadamard circuit, which is a special case of the gate teleportation protocol of Gottesman and Chuang [GC99]. This circuit is depicted in Figure 4. The magic state for this gadget is the two-qubit maximally entangled state $|00\rangle + |01\rangle + |10\rangle - |11\rangle$.

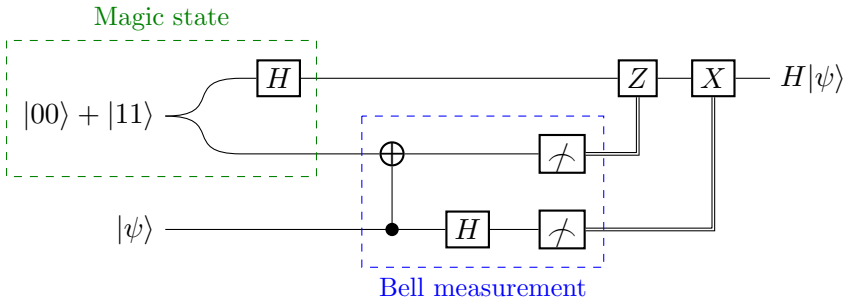


Figure 4: Teleport-through-Hadamard circuit

Implementation of the Bell measurement appearing in the above circuit requires that a Hadamard gate be applied to one of the two qubits immediately prior to measurement. At first glance this requirement might appear circular, as we require a Hadamard gate in order to implement the Hadamard gate. We claim, however, that it is possible to implement the Hadamard gate bitwise on authenticated data *provided that the qubit is measured immediately afterward* as is the case for a Bell measurement.

This claim is not difficult to justify. As mentioned above, the effect of the bitwise H gate is to swap the syndrome registers X', Z' immediately prior to measurement. But it is trivial to modify any measure-then-decode procedure (such as that mentioned in Section 5.2) so as to take this swap into account.

As with the gadget for K presented in Section 5.5.3, the correction in our gadget for H is a Pauli gate. Hence, implementation of this gadget requires only one-way classical communication from attacker to verifier.

5.6 Miscellaneous remarks

5.6.1 On the need to re-use code keys in the trap scheme

Classical keys for the trap scheme are specified by a pair $k = (\pi, P)$ indicating the choice of trap code E_π and Pauli encryption P. In Section 5.5 we saw that each gate G in our universal gate set for the trap scheme has the property that the associated gadget $\tilde{G}$ is validated by updating

only the Pauli key P so that $(\pi, P) \mapsto (\pi, P')$. This is fortunate, as our implementation of the controlled-NOT gate necessitates that every authenticated qubit share the same code key π. As noted in Section 5.2, the security of of any encode-encrypt scheme (including our trap scheme) is preserved even when the code key is re-used across multiple data registers.

The polynomial scheme of Ben-Or *et al.* has a similar structure [BOCG+06]. In particular, a key in the polynomial scheme is a pair $k = (s, P)$ consisting of a "sign key" s and a Pauli key P. The generalized controlled-NOT and Toffoli gates for the polynomial scheme necessitate that each authenticated qudit use the same sign key but different Pauli keys.

5.6.2 Clifford circuits can be implemented offline

The $\pi/8$-phase (T) gate is the only gate in the universal set described in Section 5.5 that requires two-way interaction between the attacker and verifier. Since any Clifford circuit can be implemented without T gates, it follows that any Clifford circuit can be implemented on data authenticated under the trap scheme in an *offline* manner. In particular, the transmission to the verifier of all measurement results from all gadgets for the gates in a Clifford circuit can be put off until the very end of the computation, at which time the verifier can decode the results and deduce the effect of each correction on the final Pauli key.

5.6.3 Any circuit can be implemented with only classical interaction

We can see from the gate constructions of Section 5.5 that any quantum circuit whatsoever—be it a Clifford circuit or otherwise—can be implemented on authenticated data with only *classical* interaction between attacker and verifier.

In our application to quantum one-time programs we exploit this fact in order to obtain a non-interactive protocol for two-party quantum computation by encapsulating all classical interaction inside a reactive *classical* one-time program.

The need in our quantum one-time program for a *reactive* classical one-time program is necessitated by the gadget for the T gate. In the special case that the quantum one-time program is for a Clifford circuit there is no need for a reactive classical one-time program: it suffices to use a non-reactive COTP to compute the final decryption key based on measurement results supplied by the user.

5.6.4 Rigorous security

In this section we described how gates from a universal set can be applied to data authenticated under the trap scheme. Intuitively, we can see that the interactive protocol for implementing an arbitrary circuit is cryptographically secure, yet we did not provide a fully formal proof of security in this section. If desired, a fully rigorous security proof for computation on authenticated data via the trap scheme can be obtained as a special case of the security proof for our main result on QOTPs in the next section.

6 Statement and analysis of our QOTP

In this section we will formally describe and analyze our protocol for quantum one-time programs. First, in Section 6.1, we describe a few preliminaries, in particular notation related to implementing

quantum computation using Clifford gates and magic states. In Sections 6.2 and 6.3 we specify the protocol actions for honest senders and receivers. Sections 6.4 through 6.8 contain the proof of Theorem 8, that our protocol is statistically quantum-UC-secure realization of $\mathcal{F}_{\Phi}^{\text{OTP}}$ in the case of a corrupt user, in the $\mathcal{F}^{\text{BR-OTP}}$-hybrid model. The proof makes use of a new tabular representation for operations which is given in Section 6.5 (see also Appendix C.2). We first describe the general form of the security argument in Section 6.4, analyze the environment's interaction with the sender in Section 6.6, and describe and analyze the simulator for an arbitrary adversary in Sections 6.7 and 6.8.

Our QOTP construction requires an encode-encrypt quantum authentication scheme that admits quantum computation on authenticated data, such as the trap scheme presented in Section 5. Our construction is completely independent of the specific choice of scheme. Moreover, if the underlying authentication scheme is ε-secure then our QOTP is 2ε-secure.

Henceforth we assume that such a scheme is fixed—call this scheme $\mathbf{Q}$. We let $\mathscr{E}$ denote the family of codes upon which $\mathbf{Q}$ is based and we let $\mathcal{G}$ denote the universal gate set that can be implemented on data authenticated under $\mathbf{Q}$.

6.1 Preliminaries

Suppose that we are given an arbitrary unitary V—specified as a circuit using gates from $\mathcal{G}$ and acting upon register R—and we wish to construct a circuit that implements V on data authenticated under $\mathbf{Q}$. We have already seen how to do this for the trap scheme. In this section we review this simple process so as to establish basic concepts and notation that will be useful throughout Section 6.

6.1.1 Quantum computation with Clifford gates and magic states

Let r denote the number of gates in V that require magic states. (For the trap scheme, r is the total number of K, H, and T gates in V.) Alongside the data register R we add r registers $\mathsf{M}_1, \ldots, \mathsf{M}_r$, which are assumed to be initialized to the appropriate magic states $|\mu_1\rangle, \ldots, |\mu_r\rangle$. We refer to these r registers collectively as M and to the collective state of M as $|\mu\rangle$.

Let $V^{(0)}, \ldots, V^{(r)}$ be a partition of the gates of V so that $V^{(i)}$ denotes the Clifford circuit consisting of all the gates occurring after measurement of the ith magic state and before measurement of the $(i+1)$th magic state. Note that each $V^{(i)}$ acts only upon registers $(\mathsf{R}, \mathsf{M}_{i+1})$ and $V^{(r)}$ acts only upon R. The circuit V is implemented as follows:

1. Apply $V^{(0)}$.
2. For $i = 1, \ldots, r$:
   (a) Measure the magic state register M_i and apply the Clifford correction $C^{(i)}$ indicated by that measurement result.
   (b) Apply $V^{(i)}$.

In order to see how the action of V is recovered from the above process it is helpful to write this procedure as a channel on R in Kraus form. To this end let $a \in \{0,1\}^r$ be the r-dimensional vector of classical measurement results obtained in the above implementation of V and let V_a denote the Clifford circuit resulting from applying $V^{(0)}, \ldots, V^{(r)}$ interleaved with Clifford corrections $C^{(1)}, \ldots, C^{(r)}$ according to the measurement outcomes a. Because each $V^{(i)}$ acts upon a unique

magic state register, the desired channel on R can be written as follows:

$$\rho \mapsto \sum_{a \in \{0,1\}^r} \langle a|V_a \left(\rho \otimes |\mu\rangle\langle\mu|\right) V_a^* |a\rangle \ . \tag{3}$$

It is a property of magic state implementations of gates that for each a we have

$$\langle a|V_a|\mu\rangle = \frac{1}{2^{r/2}} V \ . \tag{4}$$

So the above channel (3) is equivalent to

$$\rho \mapsto \sum_{a \in \{0,1\}^r} \frac{1}{2^r} V\rho V^* = V\rho V^* \tag{5}$$

as desired.

If for some reason the measurement results a are corrupted to some other vector a' then the above procedure will mis-apply the Clifford corrections $C^{(1)}, \ldots, C^{(r)}$. Let $V_{a-a'}$ denote the circuit derived from V by inserting extra Clifford gates according to the corruption $a - a'$ so that

$$\langle a'|V_a|\mu\rangle = \frac{1}{2^{r/2}} V_{a-a'} \ . \tag{6}$$

6.1.2 Encoded circuits

In accordance with the convention introduced in Section 5.1, any register denoted with a tilde $\tilde{}$ is assumed to be accompanied by its own X- and Z-syndrome registers. Given a logical register R and a code $E \in \mathscr{E}$, the encoded register $\tilde{\mathsf{R}}$ is obtained by applying the operator $E(I_{\mathsf{R}} \otimes |0\rangle)$ to R. For brevity we omit the initial state $|0\rangle$ of the syndrome registers and simply view E as an isometry from R to $\tilde{\mathsf{R}}$ when it is convenient to do so. When multiple registers $\mathsf{R}_1, \ldots, \mathsf{R}_k$ are each encoded under the same code E we write E instead of $E^{\otimes k}$.

Given a circuit W acting on R and composed entirely of states from $\mathcal{G}$ that can be implemented without magic states (such as the circuits V_a of the previous subsection), it is easy to construct the circuit $\tilde{W}$ acting on $\tilde{\mathsf{R}}$ that implements W on authenticated data: simply replace each logical gate with its equivalent on authenticated data.

Of course, encoding a register under a code E and then applying $\tilde{W}$ is equivalent to first applying W and then encoding the result under E. This identity is expressed succinctly under the above notation as

$$\tilde{W}E = EW.$$

6.2 Specification of the sender's message

Let $\Phi : (\mathsf{A}, \mathsf{B}) \to \mathsf{C}$ be a channel specified as a quantum circuit using gates from $\mathcal{G}$. In this section we specify the QOTP for Φ. More specifically, suppose that the input registers (A, B) are prepared in some state (possibly entangled with other registers) and given to the sender and receiver, respectively. In this section we show how to construct the sender's message to the receiver given A.

6.2.1 Implementing a channel via controlled-unitary

Without loss of generality we assume that the channel Φ has the form $\Phi : (\mathsf{A}, \mathsf{B}) \to \mathsf{B}$. That is, the receiver's output register $\mathsf{C} = \mathsf{B}$ has the same size as the input register. Furthermore, without loss of generality we assume that the channel Φ is specified by a unitary circuit U acting on registers $(\mathsf{A}, \mathsf{B}, \mathsf{E})$. The extra register E is an auxiliary register initialized to the $|0\rangle$ state. The action of Φ is recovered from U by discarding registers (A, E) so that $\Phi(\rho) = \mathrm{Tr}_{\mathsf{AE}}(U\rho U^*)$ for all ρ.

Given a circuit U one can efficiently find a circuit for the controlled-U operation, which we denote c-U. In addition to the registers $(\mathsf{A}, \mathsf{B}, \mathsf{E})$, the circuit for c-U acts upon an additional qubit called the *control qubit* so that for any pure state $|\psi\rangle$ of $(\mathsf{A}, \mathsf{B}, \mathsf{E})$ we have

$$\begin{aligned} c\text{-}U : |\psi\rangle \otimes |\mathrm{on}\rangle &\mapsto U|\psi\rangle \otimes |\mathrm{on}\rangle \\ : |\psi\rangle \otimes |\mathrm{off}\rangle &\mapsto |\psi\rangle \otimes |\mathrm{off}\rangle \quad . \end{aligned}$$

In an effort to minimize the number of distinct register names we bundle the control qubit into the ancillary register E with the understanding that the initial state of E is $|0\rangle$ for U and $|0\rangle|\mathrm{on/off}\rangle$ for c-U.

In our QOTP for Φ the sender and receiver shall implement the circuit for c-$\tilde{U}$. In this protocol the sender prepares the authenticated ancilla register $\tilde{\mathsf{E}}$ (including the control qubit) and gives it to the receiver (along with several other quantum registers and a reactive COTP to be specified shortly). This control qubit is always initialized to the $|\mathrm{on}\rangle$ state. As such, one might wonder why we bother implementing c-$\tilde{U}$ instead of $\tilde{U}$. The purpose of the control qubit is to facilitate the forthcoming security proof.

We also have an alternative QOTP in which $\tilde{U}$ is implemented directly with no need for c-$\tilde{U}$. However, the security proof for this alternative QOTP is more technically cumbersome than our protocol for c-$\tilde{U}$ (see Section 6.2.3) so we have elected to present only the protocol for c-$\tilde{U}$ in this paper. Whether or not the controlled-U is necessary for our somewhat simpler security proof is an interesting unresolved question.

6.2.2 Specification

Let r be the number of gates in c-$\tilde{U}$ that require magic states. After the parties have received their input registers A, B, a non-interactive protocol for c-$\tilde{U}$ consists of a single message from the sender to the receiver. This message consists of the following objects:

1. Quantum registers $\tilde{\mathsf{A}}, \tilde{\mathsf{B}}_{\mathsf{in}}, \tilde{\mathsf{B}}_{\mathsf{in}}, \tilde{\mathsf{B}}_{\mathsf{out}}, \tilde{\mathsf{B}}_{\mathsf{out}}, \tilde{\mathsf{E}}, \tilde{\mathsf{M}} = (\tilde{\mathsf{M}}_1, \ldots, \tilde{\mathsf{M}}_r)$ prepared in specific states described in Protocol 1 below.
2. An $(r+1)$-round reactive classical one-time program (BR-OTP) described in Protocol 2 below.

In order to prepare this message, a code $E \in \mathscr{E}$ and encryption Paulis P, S are chosen uniformly at random. The Pauli S acts on $\tilde{\mathsf{B}}_{\mathsf{out}}$ and the Pauli P acts on $(\tilde{\mathsf{A}}, \tilde{\mathsf{B}}_{\mathsf{in}}, \tilde{\mathsf{E}}, \tilde{\mathsf{M}})$. (Here and throughout the paper we adopt the convention that the portion of a multi-register Pauli acting on a single register is denoted by the register name appearing in a subscript. For example, the portion of P acting on $\tilde{\mathsf{M}}$ is denoted $P_{\tilde{\mathsf{M}}}$ and it holds that $P = P_{\tilde{\mathsf{A}}} \otimes P_{\tilde{\mathsf{B}}_{\mathsf{in}}} \otimes P_{\tilde{\mathsf{E}}} \otimes P_{\tilde{\mathsf{M}}}$.) The registers are prepared as described in Protocol 1 and Figure 5.

In addition to these registers, the sender prepares an $(r+1)$-round BR-OTP to act as described in Protocol 2.

Protocol 1 Message preparation for sender

$(\mathsf{B}_{\mathsf{in}}, \tilde{\mathsf{B}}_{\mathsf{in}})$:	Teleport-through-authentication state $P_{\tilde{\mathsf{B}}_{\mathsf{in}}} E\|\phi^+\rangle$. (See Figure 5(a).)
$(\tilde{\mathsf{B}}_{\mathsf{out}}, \mathsf{B}_{\mathsf{out}})$:	Teleport-through-de-authentication state obtained by discarding the syndrome registers of $E^* S\|\phi^+\rangle$. (See Figure 5(b).)
$\tilde{\mathsf{A}}$:	Authenticated input state. Obtained by applying $P_{\tilde{\mathsf{A}}} E$ to the input register A.
$\tilde{\mathsf{E}}$:	Authenticated ancilla $P_{\tilde{\mathsf{E}}} E\|0\rangle\|\mathrm{on}\rangle$.
$\tilde{\mathsf{M}}$:	Authenticated magic states $P_{\tilde{\mathsf{M}}} E\|\mu\rangle$ where $\|\mu\rangle = \|\mu_1\rangle \cdots \|\mu_r\rangle$ and $\|\mu_1\rangle, \ldots, \|\mu_r\rangle$ are the r magic states required for c-U.

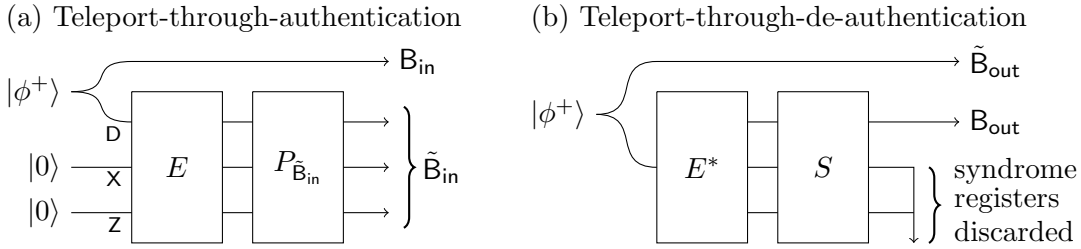


Figure 5: Circuits for teleporting through authentication and de-authentication

Protocol 2 Specification of the BR-OTP

1. Receive (a classical description of) a Pauli T^{in}.
2. For $i = 1, \ldots, r$:
   (a) Receive a classical bit string c_i. Decode c_i into a classical bit a_i as per the measure-then-decode circuit (95) of Section B.2 with code key E and decryption Pauli $\hat{P}_{\tilde{\mathsf{M}}_i}$ to be specified later. Return the decoded bit a_i to the user.
   (b) If c_i is inconsistent with $(E, \hat{P}_{\tilde{\mathsf{M}}_i})$—that is, if the measure-then-decode circuit (95) applied to c_i indicates a non-zero error syndrome—then cheating has been detected. Remember whether cheating has been detected.
3. Receive (a classical description of) a Pauli T^{out}. If cheating was never detected in step 2b then return to the user a (classical description of a) decryption Pauli $\hat{S}$ to be specified later. Otherwise return uniformly random bits. For later convenience we specify that the classical description of $\hat{S}$ be contained in a register K.

This QOTP could be mass-produced. The state of the authenticated register $\tilde{\mathsf{A}}$ depends upon the state of the sender's input register A. But the remaining registers could all be prepared (or mass-produced) before A is received. Furthermore, the BR-OTP also does not depend upon A, but it does depend upon the authentication key for $\tilde{\mathsf{A}}$. This key could be chosen in advance, in which case the BR-OTP could also be mass-produced before A is received.

6.2.3 No need to check integrity of the data registers

Notice that the BR-OTP checks the syndrome registers of *only the magic state register* $\tilde{\mathsf{M}}$ and ignores any error syndrome present in other registers. This minimal requirement might at first seem insufficient to establish a secure QOTP as, for example, the receiver is free to tamper with the data registers $(\tilde{\mathsf{A}}, \tilde{\mathsf{B}}, \tilde{\mathsf{E}})$. We shall see later that any such attack is equivalent to an attack before and/or after Φ is applied and hence can be reproduced by a simulator with one-shot access to Φ.

If instead we were to modify our BR-OTP so as to also verify the syndrome registers for the receiver's output register $\mathsf{B}_{\mathsf{out}}$ then we could derive a protocol that implements the circuit U directly, as opposed to the controlled-U implemented by the present protocol. As suggested earlier, however, the formal proof of security for such a protocol is more technically cumbersome than our current proof.

As noted in Section B.2, the measure-then-decode circuit (95) implemented by the COTP in step 2a is equivalent to the decode-then-measure circuit (96). Thus, we may equivalently assume that in step 2a the COTP applies the *quantum* circuit $\hat{P}_{\tilde{\mathsf{M}}_i} E^*$ to register $\tilde{\mathsf{M}}_i$ followed by measurements of the X-syndrome register and the data register.

6.3 Protocol for an honest receiver, completeness

The actions for an honest receiver to use the QOTP to obtain $\Phi(\rho)$ are specified in Protocol 3.

6.4 General form of an arbitrary environment

UC security of the protocol of Section 6.2 is proved as follows. First, consider an arbitrary (possibly cheating) receiver who receives the message from the sender and interacts with the sender's BR-OTP. Throughout the interaction the receiver also exchanges messages with an *environment*. UC security is established by exhibiting a simulator that mimics the behaviour of the receiver from the environment's point of view using only calls to the ideal functionality. Specifically, the environment selects the input registers (A, B) for both sender and receiver, interacts with either the real receiver or the simulated receiver, then produces a single bit indicating the environment's guess as to whether it interacted with the real or simulated receiver.

By the completeness of dummy adversaries [Can01, Unr10], it suffices to assume that the receiver simply shuttles messages between the environment and the honest sender. For convenience this dummy receiver can be absorbed into the environment, leaving only an interaction between the environment and the honest sender (or simulated honest sender). In this section we write down a general form that every such environment must have.

Given the discussion above, we assume without loss of generality that the actions of any environment throughout the interaction are as described in Protocol 4.

Let us now argue that no generality is lost in assuming an environment described in Protocol 4.

Protocol 3 Honest use of QOTP

1. Perform a Bell measurement on $(\mathsf{B}, \mathsf{B}_{\mathsf{in}})$ so as to teleport-through-authentication. This is achieved by first applying the unitary Bell rotation B followed by a measurement $\{|T^{\mathsf{in}}\rangle\langle T^{\mathsf{in}}|\}$ where each $|T^{\mathsf{in}}\rangle\langle T^{\mathsf{in}}|$ is a projector onto the classical basis state indicating Pauli correction T^{in}. The receiver provides T^{in} as the first input to the BR-OTP.
   At this time the contents of B have been authenticated and placed in register $\tilde{\mathsf{B}}_{\mathsf{in}}$.
2. Run the protocol of Section 6.1.1 with $V = c\text{-}\tilde{U}$ so as to apply $c\text{-}\tilde{U}$ to the authenticated registers $(\tilde{\mathsf{A}}, \tilde{\mathsf{B}}_{\mathsf{in}}, \tilde{\mathsf{E}}, \tilde{\mathsf{M}})$. Explicitly,
   (a) Apply $c\text{-}\tilde{U}^{(0)}$.
   (b) For $i = 1, \ldots, r$:
      i. Measure the magic state register $\tilde{\mathsf{M}}_i$ and provide the result as input to the BR-OTP.
      ii. The BR-OTP provides as output a single bit indicating whether to apply the associated Clifford correction $\tilde{C}^{(i)}$.
      iii. Apply $c\text{-}\tilde{U}^{(i)}$.

   The implementation of $c\text{-}\tilde{U}$ is now complete. At this time the register $(\tilde{\mathsf{A}}, \tilde{\mathsf{B}}_{\mathsf{in}}, \tilde{\mathsf{E}})$ holds the authenticated version of $(\mathsf{A}, \mathsf{B}, \mathsf{E})$ with $c\text{-}U$ applied.
3. Perform a Bell measurement on $(\tilde{\mathsf{B}}_{\mathsf{in}}, \tilde{\mathsf{B}}_{\mathsf{out}})$ so as to teleport-through-de-authentication. As above, the result of this measurement indicates a Pauli correction T^{out}, which the receiver provides as the final input to the BR-OTP.
   At this time the register $\mathsf{B}_{\mathsf{out}}$ holds the receiver's output. This register is encrypted but not authenticated.
4. For its final output, the BR-OTP provides the Pauli decryption key $\hat{S}$ to be specified later. Apply this Pauli to $\mathsf{B}_{\mathsf{out}}$ to recover the receiver's output.

Protocol 4 General form of an arbitrary environment

1. Prepare registers $(\mathsf{A}, \mathsf{B}, \mathsf{W})$ in an arbitrary pure state $|\psi\rangle$ and provide A to the sender (or simulated sender) as input.
2. Receive from the sender (or simulated sender) quantum registers $(\tilde{\mathsf{A}}, \mathsf{B}_{\mathsf{in}}, \tilde{\mathsf{B}}_{\mathsf{in}}, \mathsf{B}_{\mathsf{out}}, \tilde{\mathsf{B}}_{\mathsf{out}}, \tilde{\mathsf{E}}, \tilde{\mathsf{M}})$. Apply a unitary $K^{(0)}$ to all the registers, then perform a Bell measurement on $(\mathsf{B}, \mathsf{B}_{\mathsf{in}})$ so as to teleport-through-authentication. Provide the resulting Pauli T^{in} as the first input to the BR-OTP, just as an honest receiver would.
3. For $i = 1, \ldots, r$:
   (a) Provide the register $\tilde{\mathsf{M}}_i$ to the BR-OTP.
   (b) The BR-OTP returns a single bit a_i.
   (c) Apply a unitary $K_{a_i}^{(i)}$ to the remaining registers. (That is, every register except $\mathsf{B}, \mathsf{B}_{\mathsf{in}}, \tilde{\mathsf{M}}_1, \ldots, \tilde{\mathsf{M}}_i$.) The subscript a_i indicates that the environment's choice of $K_{a_i}^{(i)}$ could depend upon the bit a_i.
4. Perform a Bell measurement on $(\tilde{\mathsf{B}}_{\mathsf{in}}, \tilde{\mathsf{B}}_{\mathsf{out}})$ so as to teleport-through-de-authentication. Provide the resulting Pauli T^{out} as the final input to the BR-OTP, just as an honest receiver would.
5. The BR-OTP provides as its final output (a classical description of) a decryption Pauli $\hat{S}$ to be specified later, stored in a new register K.
6. Perform a binary-valued measurement on the remaining registers $\tilde{\mathsf{A}}, \mathsf{B}_{\mathsf{out}}, \mathsf{E}, \mathsf{W}, \mathsf{K}$.

1. The BR-OTP has by definition classical input/output behaviour that does not preserve coherent superpositions of classical basis states. As such, any qubits touched by the BR-OTP are assumed to be measured in the computational basis. Thus, it makes no difference whether data sent to a BR-OTP is measured by the environment or by the BR-OTP, so we are free to assume that any given measurement is performed by whichever party better suits our discussion.
2. In steps 2 and 4 it is assumed that the environment does a proper Bell measurement of $(\mathsf{B}, \mathsf{B}_{\mathsf{in}})$ and $(\tilde{\mathsf{B}}_{\mathsf{in}}, \tilde{\mathsf{B}}_{\mathsf{out}})$, respectively, and then faithfully reports the results to the BR-OTP just as an honest receiver would. This assumption is justified because any tampering the environment might wish to do with the measurements or the results thereof can instead be incorporated into the environment's circuits $K_{a_i}^{(i)}$.
   For example, an environment who wishes to tamper with the Bell measurement in step 2 could select the circuit $K^{(0)}$ so as to prepare the registers $(\mathsf{B}, \mathsf{B}_{\mathsf{in}})$ in any desired state by swapping out data from the memory workspace W (or even other registers from the sender) and then pre-invert the rotation into the Bell basis prior to measurement. Substituting this choice of $K^{(0)}$ into Protocol 4, we see that such an environment undoes the Bell rotation and thus the resulting measurement of $(\mathsf{B}, \mathsf{B}_{\mathsf{in}})$ reproduces exactly the result intended by the environment.
3. In step 3a it is assumed that the environment simply transmits the unmeasured register $\tilde{\mathsf{M}}_i$ to the BR-OTP.
   As explained above, we are free to make this assumption as the BR-OTP has purely classical input/output behaviour. This assumption is convenient because it allows us to seamlessly substitute the measure-then-decode procedure with its equivalent decode-then-measure procedure acting on unmeasured quantum data as described in Section 5.2.
4. Intuitively, the environment's ith circuit $K_{a_i}^{(i)}$ could depend upon all prior classical data exchanged with the BR-OTP. Yet in step 3c we explicitly allow only for dependence upon the most recent bit a_i received from the BR-OTP. Furthermore, we make the simplifying assumption that each $K_{a_i}^{(i)}$ does not act upon the registers $\mathsf{B}, \mathsf{B}_{\mathsf{in}}, \tilde{\mathsf{M}}_1, \ldots, \tilde{\mathsf{M}}_{i-1}$ that were measured earlier in the protocol. No generality is lost, however, because prior circuits are free to copy classical data into fresh space within the workspace register W for reference in future rounds.

6.5 A tabular representation for operators and vectors

Security of our QOTP is established by analysis of the state of all the registers in the environment's posession immediately prior to the final binary-valued measurement in step 6 of Protocol 4. However, it is difficult to produce a concise description of this state due to the large number of distinct registers and the many rounds of interaction with the BR-OTP. To combat this difficulty we introduce a tabular representation for operators and vectors.

This new way of describing states is best explained by example. To this end recall that the initial state of registers $(\mathsf{A}, \mathsf{B}, \mathsf{W})$ is $|\psi\rangle$ and that register A is given to the sender as input. The sender introduces registers $\mathsf{B}_{\mathsf{in}}, \tilde{\mathsf{B}}_{\mathsf{in}}, \tilde{\mathsf{B}}_{\mathsf{out}}, \mathsf{B}_{\mathsf{out}}, \tilde{\mathsf{E}}, \tilde{\mathsf{M}}$. For each choice of code key E, encryption keys P, S, Bell measurement results $T^{\mathsf{in}}, T^{\mathsf{out}}$, and magic state data- and syndrome-measurement results $a = (a_1, \ldots, a_r), s = (s_1, \ldots, s_r)$, the unnormalized pure state of the remaining (unmeasured) quantum registers $(\mathsf{B}_{\mathsf{out}}, \tilde{\mathsf{A}}, \mathsf{W}, \tilde{\mathsf{E}})$ after step 4 of Protocol 4 (that is, immediately before the BR-

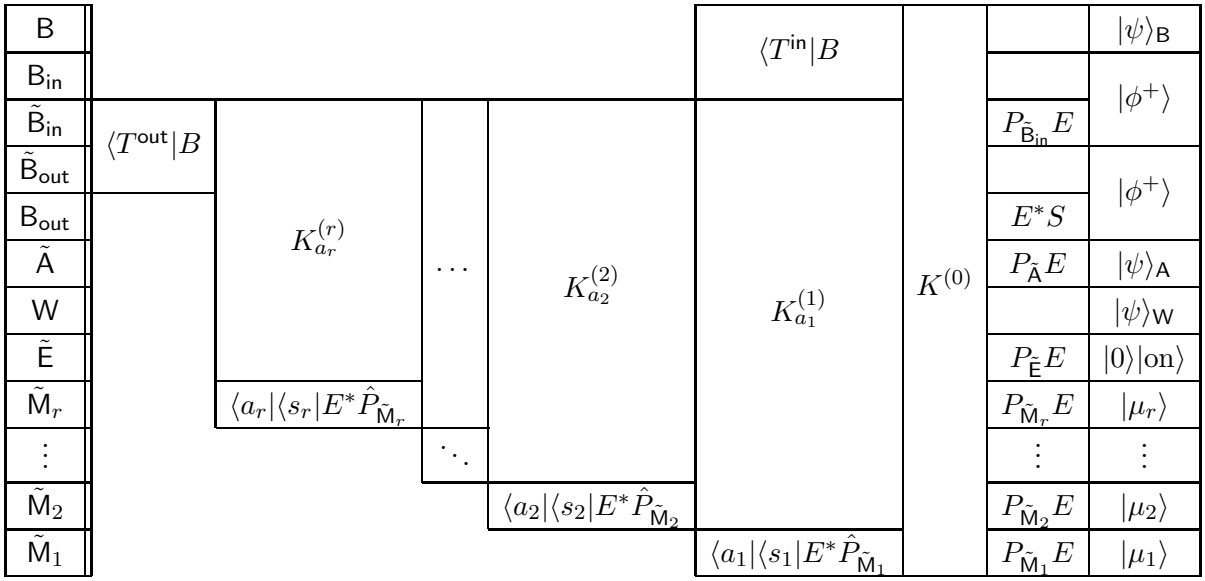


Figure 6: Tabular representation of the state of the system after step 4 of Protocol 4.

OTP reveals the decryption key register K) is given by the table in Figure 6.

In general, rows in a table correspond to registers in the joint system. Cells within each row correspond to individual operators or vectors and are ordered from right to left as per the convention for operator composition. Empty cells implicitly indicate the identity operator. The table as a whole specifies an operator (or vector) as a composition of product operators on the registers.

In this particular table we have used the notation $|\psi\rangle_{\mathsf{A}}, |\psi\rangle_{\mathsf{B}}, |\psi\rangle_{\mathsf{W}}$ to refer to the portions of $|\psi\rangle$ contained in registers $\mathsf{A}, \mathsf{B}, \mathsf{W}$, respectively.

6.6 Analysis of the environment's interaction with the real sender

At the end of its interaction the environment produces a single output bit. In order to prove UC security we must exhibit a simulator that interacts with the environment such that the distribution on the environment's output bit is nearly identical to that induced by the environment's interaction with the real sender and BR-OTP.

We accomplish this task by deriving an expression for the state ρ_{real} of the registers $(\mathsf{B}_{\mathsf{out}}, \tilde{\mathsf{A}}, \tilde{\mathsf{E}}, \mathsf{W}, \mathsf{K})$ in the environment's possession at the end of step 5 of Protocol 4. In Section 6.7 we will exhibit a simulator, then in Section 6.8 we derive an expression for the associated state ρ_{sim} at the end of the simulated interaction and show that the trace distance between ρ_{real} and ρ_{sim} is negligible, from which the security of our QOTP follows.

Recall that in step 5 the register K is revealed by the BR-OTP. Thus, if the state of the system at the end of step 5 is ρ_{real} then state of the system at the end of step 4 must be $\mathrm{Tr}_{\mathsf{K}}(\rho_{\text{real}})$. We begin our analysis by focussing on the state $\mathrm{Tr}_{\mathsf{K}}(\rho_{\text{real}})$.

Let $|\mathscr{P}|, |\mathscr{S}|$ denote the number of Pauli operations acting on $(\tilde{\mathsf{B}}_{\mathsf{in}}, \tilde{\mathsf{M}}, \tilde{\mathsf{P}}), \tilde{\mathsf{B}}_{\mathsf{out}}$, respectively, so that each encryption Pauli P, S is chosen with probability $1/|\mathscr{P}|, 1/|\mathscr{S}|$, respectively. The

normalized mixed state $\mathrm{Tr}_{\mathsf{K}}(\rho_{\mathrm{real}})$ of $(\mathsf{B}_{\mathsf{out}}, \tilde{\mathsf{A}}, \mathsf{W}, \tilde{\mathsf{E}})$ at step 4 is

$$\mathrm{Tr}_{\mathsf{K}}(\rho_{\mathrm{real}}) = \frac{1}{|\mathscr{E}||\mathscr{P}||\mathscr{S}|} \sum_{E,P,S,a,s,T^{\mathsf{in}},T^{\mathsf{out}}} |(6)\rangle\langle(6)| \tag{7}$$

where $|(6)\rangle$ denotes the unnormalized pure state depicted in Figure 6.

This state can be written much more succinctly. Let $a = (a_1, \ldots, a_r)$ and $s = (s_1, \ldots, s_r)$ denote the vectors of magic-state data and syndrome values and write

$$K_a \stackrel{\mathrm{def}}{=} K^{(r)}_{a_r} \cdots K^{(1)}_{a_1} K^{(0)} \tag{8}$$
$$\langle a| \stackrel{\mathrm{def}}{=} \langle a_r| \cdots \langle a_1| \tag{9}$$
$$\langle s| \stackrel{\mathrm{def}}{=} \langle s_r| \cdots \langle s_1|. \tag{10}$$

The rows of table (6) corresponding to the magic state registers $\tilde{\mathsf{M}}_1, \ldots, \tilde{\mathsf{M}}_r$ can be amalgamated into a single row $\tilde{\mathsf{M}}$. The table (6) can be written

$|(6)\rangle =$

B	$\langle T^{\mathsf{in}}\|B$			$\|\psi\rangle_{\mathsf{B}}$
B_{in}				$\|\phi^+\rangle$
$\tilde{\mathsf{B}}_{\mathsf{in}}$	$\langle T^{\mathsf{out}}\|B$		$P_{\tilde{\mathsf{B}}_{\mathsf{in}}} E$	
$\tilde{\mathsf{B}}_{\mathsf{out}}$				$\|\phi^+\rangle$
$\mathsf{B}_{\mathsf{out}}$		K_a	$E^* S$	
$\tilde{\mathsf{A}}$			$P_{\tilde{\mathsf{A}}} E$	$\|\psi\rangle_{\mathsf{A}}$
W				$\|\psi\rangle_{\mathsf{W}}$
$\tilde{\mathsf{E}}$			$P_{\tilde{\mathsf{E}}} E$	$\|0\rangle\|\mathrm{on}\rangle$
$\tilde{\mathsf{M}}$	$\langle a\|\langle s\|E^* \hat{P}_{\tilde{\mathsf{M}}}$		$P_{\tilde{\mathsf{M}}} E$	$\|\mu\rangle$

(11)

Insert a superfluous $I = (c\text{-}\tilde{U}_a)^*(c\text{-}\tilde{U}_a)$ acting on registers $(\tilde{\mathsf{B}}_{\mathsf{in}}, \tilde{\mathsf{A}}, \tilde{\mathsf{E}}, \tilde{\mathsf{M}})$ into the table (11) immediately prior to K_a and reorder the rows to get

$|(6)\rangle =$

W						$\|\psi\rangle_{\mathsf{W}}$
B	$\langle T^{\mathsf{in}}\|B$					$\|\psi\rangle_{\mathsf{B}}$
B_{in}						$\|\phi^+\rangle$
$\tilde{\mathsf{B}}_{\mathsf{in}}$	$\langle T^{\mathsf{out}}\|B$		$(c\text{-}\tilde{U}_a)^*_{\tilde{\mathsf{B}}_{\mathsf{in}}}$	$(c\text{-}\tilde{U}_a)_{\tilde{\mathsf{B}}_{\mathsf{in}}}$	$P_{\tilde{\mathsf{B}}_{\mathsf{in}}} E$	
$\tilde{\mathsf{B}}_{\mathsf{out}}$						$\|\phi^+\rangle$
$\mathsf{B}_{\mathsf{out}}$		K_a			$E^* S$	
$\tilde{\mathsf{A}}$			$(c\text{-}\tilde{U}_a)^*_{\tilde{\mathsf{A}}}$	$(c\text{-}\tilde{U}_a)_{\tilde{\mathsf{A}}}$	$P_{\tilde{\mathsf{A}}} E$	$\|\psi\rangle_{\mathsf{A}}$
$\tilde{\mathsf{E}}$			$(c\text{-}\tilde{U}_a)^*_{\tilde{\mathsf{E}}}$	$(c\text{-}\tilde{U}_a)_{\tilde{\mathsf{E}}}$	$P_{\tilde{\mathsf{E}}} E$	$\|0\rangle\|\mathrm{on}\rangle$
$\tilde{\mathsf{M}}$	$\langle a\|\langle s\|E^* \hat{P}_{\tilde{\mathsf{M}}}$		$(c\text{-}\tilde{U}_a)^*_{\tilde{\mathsf{M}}}$	$(c\text{-}\tilde{U}_a)_{\tilde{\mathsf{M}}}$	$P_{\tilde{\mathsf{M}}} E$	$\|\mu\rangle$

(12)

(Here we have used the notation $(c\text{-}\tilde{U}_a)^*_{\tilde{\mathsf{A}}}$, *etc.* to illustrate the fact that the unitary $(c\text{-}\tilde{U}_a)^*$ is split over several nonadjacent rows in the table. If the rows $\tilde{\mathsf{A}}, \tilde{\mathsf{B}}_{\mathsf{in}}, \tilde{\mathsf{E}}, \tilde{\mathsf{M}}$ were all adjacent then this

notation would be unnecessary; all four of those cells could be merged into a single larger cell for the operator $(c\text{-}\tilde{U}_a)^*$. Unfortunately, it is not possible to order the rows so as to eliminate the need to split some operators or states across nonadjacent rows.)

At an intuitive level, if the environment is to avoid being detected as a cheater by the BR-OTP then whatever the environment does in K_a must cancel out $(c\text{-}\tilde{U}_a)^*$ so that $c\text{-}\tilde{U}_a$ is implemented on $(\tilde{\mathsf{B}}_{\mathsf{in}}, \tilde{\mathsf{A}}, \tilde{\mathsf{E}}, \tilde{\mathsf{M}})$.

Let

$$K_a(c\text{-}\tilde{U}_a)^* = \sum_{\text{Paulis } Q} \alpha_Q Q \tag{13}$$

be a Pauli decomposition of $K_a(c\text{-}\tilde{U}_a)^*$. Substitute this decomposition into (12) to obtain

$$|(6)\rangle = \sum_{\text{Paulis } Q} \alpha_Q |(15)\rangle \tag{14}$$

where the table (15) is given by

W		$Q_{\tilde{\mathsf{W}}}$			$\|\psi\rangle_{\mathsf{W}}$
B	$\langle T^{\mathsf{in}}\|B$	Q_{B}			$\|\psi\rangle_{\mathsf{B}}$
B_{in}		$Q_{\mathsf{B}_{\mathsf{in}}}$			$\|\phi^+\rangle$
$\tilde{\mathsf{B}}_{\mathsf{in}}$	$\langle T^{\mathsf{out}}\|B$	$Q_{\tilde{\mathsf{B}}_{\mathsf{in}}}$	$(c\text{-}\tilde{U}_a)_{\tilde{\mathsf{B}}_{\mathsf{in}}}$	$P_{\tilde{\mathsf{B}}_{\mathsf{in}}} E$	
$\tilde{\mathsf{B}}_{\mathsf{out}}$		$Q_{\tilde{\mathsf{B}}_{\mathsf{out}}}$			$\|\phi^+\rangle$
$\mathsf{B}_{\mathsf{out}}$		$Q_{\mathsf{B}_{\mathsf{out}}}$		$E^* S$	
$\tilde{\mathsf{A}}$		$Q_{\tilde{\mathsf{A}}}$	$(c\text{-}\tilde{U}_a)_{\tilde{\mathsf{A}}}$	$P_{\tilde{\mathsf{A}}} E$	$\|\psi\rangle_{\mathsf{A}}$
$\tilde{\mathsf{E}}$		$Q_{\tilde{\mathsf{E}}}$	$(c\text{-}\tilde{U}_a)_{\tilde{\mathsf{E}}}$	$P_{\tilde{\mathsf{E}}} E$	$\|0\rangle\|\text{on}\rangle$
$\tilde{\mathsf{M}}$	$\langle a\|\langle s\|E^* \hat{P}_{\tilde{\mathsf{M}}}$	$Q_{\tilde{\mathsf{M}}}$	$(c\text{-}\tilde{U}_a)_{\tilde{\mathsf{M}}}$	$P_{\tilde{\mathsf{M}}} E$	$\|\mu\rangle$

(15)

Let us consider the teleportation operations. According to Appendix C, the rows for B and B_{in} can be folded into $\tilde{\mathsf{B}}_{\mathsf{in}}$ to obtain

$$|(15)\rangle = |\mathsf{B}|^{-1/2}$$

$\tilde{\mathsf{B}}_{\mathsf{in}}$	$\langle T^{\mathsf{out}}\|B$	$Q_{\tilde{\mathsf{B}}_{\mathsf{in}}} (c\text{-}\tilde{U}_a)_{\tilde{\mathsf{B}}_{\mathsf{in}}} P_{\tilde{\mathsf{B}}_{\mathsf{in}}} E Q^{\mathsf{T}}_{\mathsf{B}_{\mathsf{in}}} T^{\mathsf{in}} Q_{\mathsf{B}}$		$\|\psi\rangle_{\mathsf{B}}$
$\tilde{\mathsf{B}}_{\mathsf{out}}$		$Q_{\tilde{\mathsf{B}}_{\mathsf{out}}}$		$\|\phi^+\rangle$
$\mathsf{B}_{\mathsf{out}}$		$Q_{\mathsf{B}_{\mathsf{out}}}$	$E^* S$	
Registers $\tilde{\mathsf{A}}, \tilde{\mathsf{E}}, \tilde{\mathsf{M}}, \mathsf{W}$ same as (15).				

(16)

(Here $|\mathsf{R}|$ denotes the dimension of the space associated with register R.) We can then fold $\tilde{\mathsf{B}}_{\mathsf{in}}, \tilde{\mathsf{B}}_{\mathsf{out}}$ into $\mathsf{B}_{\mathsf{out}}$ so as to obtain

$$|(15)\rangle = |\mathsf{B}|^{-1/2} |\tilde{\mathsf{B}}_{\mathsf{in}}|^{-1/2}$$

$\mathsf{B}_{\mathsf{out}}$	$Q_{\mathsf{B}_{\mathsf{out}}} E^* S Q^{\mathsf{T}}_{\tilde{\mathsf{B}}_{\mathsf{out}}} T^{\mathsf{out}} Q_{\tilde{\mathsf{B}}_{\mathsf{in}}} (c\text{-}\tilde{U}_a)_{\tilde{\mathsf{B}}_{\mathsf{in}}} P_{\tilde{\mathsf{B}}_{\mathsf{in}}} E Q^{\mathsf{T}}_{\mathsf{B}_{\mathsf{in}}} T^{\mathsf{in}} Q_{\mathsf{B}}$	$\|\psi\rangle_{\mathsf{B}}$
Registers $\tilde{\mathsf{A}}, \tilde{\mathsf{E}}, \tilde{\mathsf{M}}, \mathsf{W}$ same as (15).		

(17)

Notice the normalization factors $|\mathsf{B}|^{-1/2}, |\tilde{\mathsf{B}}_{\mathsf{in}}|^{-1/2}$ picked up in the process—we will include those later. Let us re-write the entire table, amalgamating $c\text{-}\tilde{U}_a$, P, and $|\psi\rangle$.

$$|(15)\rangle = \left|\begin{array}{|c||c|c|c|c|c|c|c|}
\hline
\mathsf{W} & & Q_{\mathsf{W}} & & & & & \\
\hline
\mathsf{B}_{\mathsf{out}} & & Q_{\mathsf{B}_{\mathsf{out}}} E^* S Q_{\tilde{\mathsf{B}}_{\mathsf{out}}}^{\mathsf{T}} T^{\mathsf{out}} Q_{\tilde{\mathsf{B}}_{\mathsf{in}}} & & & E & Q_{\mathsf{B}_{\mathsf{in}}}^{\mathsf{T}} T^{\mathsf{in}} Q_{\mathsf{B}} & |\psi\rangle \\
\hline
\tilde{\mathsf{A}} & & Q_{\tilde{\mathsf{A}}} & c\text{-}\tilde{U}_a & P & E & & \\
\hline
\tilde{\mathsf{E}} & & Q_{\tilde{\mathsf{E}}} & & & E & & |0\rangle|\mathrm{on}\rangle \\
\hline
\tilde{\mathsf{M}} & \langle a|\langle s|E^* \hat{P}_{\tilde{\mathsf{M}}} & Q_{\tilde{\mathsf{M}}} & & & E & & |\mu\rangle \\
\hline
\end{array}\right| \tag{18}$$

The next step is to use commutation relations so as to massage this table into a nice form for the Pauli sandwich (Appendix B).

To this end let us first dispense with the annoying transposition operations appearing on $Q_{\tilde{\mathsf{B}}_{\mathsf{out}}}^{\mathsf{T}}, Q_{\mathsf{B}_{\mathsf{in}}}^{\mathsf{T}}$. Note that for any multi-qubit Pauli operator P the transpose $P^{\mathsf{T}} = \pm P$ with the negative phase occurring whenever P is a product of an odd number of Y-Paulis. Let $y(P) \in \{\pm 1\}$ denote this phase so that $P^{\mathsf{T}} = y(P)P$ for all Paulis P. We apply this identity to the above table, producing an extra phase factor $y(Q_{\tilde{\mathsf{B}}_{\mathsf{out}}}) y(Q_{\mathsf{B}_{\mathsf{in}}})$.

We require some notation before proceeding further. For any Paulis P, Q we write

$$c(P,Q) \stackrel{\text{def}}{=} \begin{cases} +1 & \text{if } PQ = QP \\ -1 & \text{if } PQ = -QP \end{cases} \tag{19}$$

for the phase picked up by commuting P, Q. By analogy to the notation introduced in Section 5.1, for any code E and any Pauli P we let P_E denote the Pauli with $P_E E = EP$. Recall that for any vector a of measurement results the encoded circuit $c\text{-}\tilde{U}_a$ is a Clifford circuit. As such, for any Pauli P we let $\pi_a(P)$ denote the Pauli with

$$\pi_a(P)(c\text{-}\tilde{U}) = (c\text{-}\tilde{U}_a)P \ . \tag{20}$$

Finally, recall that $(c\text{-}\tilde{U}_a)E = E(c\text{-}U_a)$ as noted in Section 6.1.2.

Now, concentrate only on the rightmost four columns of the table (18):

1. $Q_{\mathsf{B}_{\mathsf{in}}}$ commutes with T^{in}, picking up a phase of $c(T^{\mathsf{in}}, Q_{\mathsf{B}_{\mathsf{in}}})$.

$$\left|\begin{array}{|c|c|c|c|}
\hline
c\text{-}\tilde{U}_a & P & E & T^{\mathsf{in}} Q_{\mathsf{B}_{\mathsf{in}}} Q_{\mathsf{B}} \\
\hline
 & & E & \\
\hline
 & & E & \\
\hline
 & & E & \\
\hline
\end{array}\right| \tag{21}$$

2. Replace ET^{in} with $T_E^{\mathsf{in}} E$.

$$\left|\begin{array}{|c|c|c|c|c|}
\hline
c\text{-}\tilde{U}_a & P & T_E^{\mathsf{in}} & E & Q_{\mathsf{B}_{\mathsf{in}}} Q_{\mathsf{B}} \\
\hline
 & & & E & \\
\hline
 & & & E & \\
\hline
 & & & E & \\
\hline
\end{array}\right| \tag{22}$$

3. Since $c\text{-}\tilde{U}_a$ is a Clifford, we can pull PT_E^{in} through $c\text{-}\tilde{U}_a$.

$\pi_a(PT_E^{\mathsf{in}})$	$c\text{-}\tilde{U}_a$	E	$Q_{\mathsf{B}_{\mathsf{in}}}Q_{\mathsf{B}}$
		E	
		E	
		E	

(23)

4. Then we can pull E through $c\text{-}\tilde{U}_a$.

$\pi_a(PT_E^{\mathsf{in}})$	E	$c\text{-}U_a$	$Q_{\mathsf{B}_{\mathsf{in}}}Q_{\mathsf{B}}$
	E		
	E		
	E		

(24)

The entire table is now

$|(15)\rangle =$

W		Q_{W}					$\|\psi\rangle$
$\mathsf{B}_{\mathsf{out}}$		$Q_{\mathsf{B}_{\mathsf{out}}}E^*SQ_{\tilde{\mathsf{B}}_{\mathsf{out}}}T^{\mathsf{out}}Q_{\tilde{\mathsf{B}}_{\mathsf{in}}}$	$\pi_a(PT_E^{\mathsf{in}})$	E	$c\text{-}U_a$	$Q_{\mathsf{B}_{\mathsf{in}}}Q_{\mathsf{B}}$	
$\tilde{\mathsf{A}}$		$Q_{\tilde{\mathsf{A}}}$		E			
$\tilde{\mathsf{E}}$		$Q_{\tilde{\mathsf{E}}}$		E			$\|0\rangle\|\text{on}\rangle$
$\tilde{\mathsf{M}}$	$\langle a,s\|E^*\hat{P}_{\tilde{\mathsf{M}}}$	$Q_{\tilde{\mathsf{M}}}$		E			$\|\mu\rangle$

(25)

We now have an expression for the decryption Pauli $\hat{P}_{\tilde{\mathsf{M}}}$ used by the BR-OTP:

$$\hat{P}_{\tilde{\mathsf{M}}} \stackrel{\text{def}}{=} \pi_a(PT_E^{\mathsf{in}})_{\tilde{\mathsf{M}}}. \tag{26}$$

In order to clean up the extra phases we introduced by the above commutation relations we write

$$\alpha_{Q,T^{\mathsf{in}}} \stackrel{\text{def}}{=} y(Q_{\tilde{\mathsf{B}}_{\mathsf{out}}})y(Q_{\mathsf{B}_{\mathsf{in}}})c(T^{\mathsf{in}},Q_{\mathsf{B}_{\mathsf{in}}})\alpha_Q \tag{27}$$

for each choice of Paulis Q, T^{in} so that the original state (6) can be written

$$|(6)\rangle = |\mathsf{B}|^{-1/2}|\tilde{\mathsf{B}}_{\mathsf{in}}|^{-1/2}\sum_{\text{Paulis } Q}\alpha_{Q,T^{\mathsf{in}}}|(25)\rangle. \tag{28}$$

Until now we have used only simple commutation relations to derive an alternate expression for the state $|(15)\rangle$. Now we wish to employ the Pauli sandwich on register $\tilde{\mathsf{M}}$ so that the double sum over $Q_{\tilde{\mathsf{M}}}$ becomes a single sum. To this end consider the table

W	Q_{W}					$\|\psi\rangle$
$\mathsf{B}_{\mathsf{out}}$	$Q_{\mathsf{B}_{\mathsf{out}}}E^*SQ_{\tilde{\mathsf{B}}_{\mathsf{out}}}T^{\mathsf{out}}Q_{\tilde{\mathsf{B}}_{\mathsf{in}}}$	$\pi_a(PT_E^{\mathsf{in}})_{\mathsf{B}_{\mathsf{out}}}$	E	$c\text{-}U_a$	$Q_{\mathsf{B}_{\mathsf{in}}}Q_{\mathsf{B}}$	
$\tilde{\mathsf{A}}$	$Q_{\tilde{\mathsf{A}}}$	$\pi_a(PT_E^{\mathsf{in}})_{\tilde{\mathsf{A}}}$	E			
$\tilde{\mathsf{E}}$	$Q_{\tilde{\mathsf{E}}}$	$\pi_a(PT_E^{\mathsf{in}})_{\tilde{\mathsf{E}}}$	E			$\|0\rangle\|\text{on}\rangle$
$\tilde{\mathsf{M}}$						$\|\mu\rangle$

(29)

obtained from (25) by deleting everything in row $\tilde{\mathsf{M}}$ occurring after the operator $c\text{-}U_a$. For each Pauli $Q_{\tilde{\mathsf{M}}}$ define

$$|Q_{\tilde{\mathsf{M}}}\rangle \stackrel{\text{def}}{=} \sum_{\neg Q_{\tilde{\mathsf{M}}}} \alpha_{Q,T_{\mathsf{in}}} |(29)\rangle. \tag{30}$$

Here the summation over $\neg Q_{\tilde{\mathsf{M}}}$ is shorthand for a summation over all Paulis Q_{B}, $Q_{\mathsf{B}_{\mathsf{in}}}$, $Q_{\tilde{\mathsf{B}}_{\mathsf{in}}}$, $Q_{\tilde{\mathsf{B}}_{\mathsf{out}}}$, $Q_{\mathsf{B}_{\mathsf{out}}}$, $Q_{\tilde{\mathsf{A}}}$, $Q_{\tilde{\mathsf{E}}}$, Q_{W}—that is, all Paulis comprising Q except the Pauli $Q_{\tilde{\mathsf{M}}}$ acting on $\tilde{\mathsf{M}}$. Note that the only dependence of $|Q_{\tilde{\mathsf{M}}}\rangle$ on $Q_{\tilde{\mathsf{M}}}$ is in the scalar $\alpha_{Q,T^{\mathsf{in}}}$. Keep in mind that $|Q_{\tilde{\mathsf{M}}}\rangle$ also depends upon T^{in}, T^{out}, a, s, E, $\pi_a(PT^{\mathsf{in}}_E)_{\tilde{\mathsf{B}}_{\mathsf{out}}}$, $\pi_a(PT^{\mathsf{in}}_E)_{\tilde{\mathsf{A}}}$, $\pi_a(PT^{\mathsf{in}}_E)_{\tilde{\mathsf{E}}}$, S; we have simply omitted these parameters for brevity.

By the Pauli sandwich (Lemma 13) we have that the mixed state $\mathrm{Tr}_{\mathsf{K}}(\rho_{\text{real}})$ from (7) equals

$$\mathrm{Tr}_{\mathsf{K}}(\rho_{\text{real}}) = \frac{1}{|\mathscr{E}||\mathscr{P}||\mathscr{S}||\mathsf{B}||\tilde{\mathsf{B}}_{\mathsf{in}}|} \sum_{E,P,S,a,s,T^{\mathsf{in}},T^{\mathsf{out}},Q_{\tilde{\mathsf{M}}}} \langle a|\langle s|E^* Q_{\tilde{\mathsf{M}}} E|Q_{\tilde{\mathsf{M}}}\rangle\langle Q_{\tilde{\mathsf{M}}}|E^* Q_{\tilde{\mathsf{M}}} E|a\rangle|s\rangle \tag{31}$$

From here consider two cases corresponding to zero and non-zero syndrome measurements ($s = 0, s \neq 0$), indicating acceptance and rejection by the BR-OTP, respectively. In so doing we decompose ρ_{real} into a sum of three positive semidefinite operators

$$\rho_{\text{real}} = R_{\text{rej}} + R_{\text{acc}} + [\varepsilon_{\text{real}}] \tag{32}$$

where $[\varepsilon_{\text{real}}]$ has trace at most ε. Then in Section 6.8 we will show that

$$\rho_{\text{sim}} = R_{\text{rej}} + R_{\text{acc}} + [\varepsilon_{\text{sim}}] \tag{33}$$

for some $[\varepsilon_{\text{sim}}]$ that has trace at most ε, from which it will follow that

$$\|\rho_{\text{real}} - \rho_{\text{sim}}\|_{\text{Tr}} = \|[\varepsilon_{\text{real}}] - [\varepsilon_{\text{sim}}]\|_{\text{Tr}} \leq 2\varepsilon \tag{34}$$

as desired.

6.6.1 In the event of acceptance

Consider the unnormalized mixed state obtained from the expression (31) for $\mathrm{Tr}_{\mathsf{K}}(\rho_{\text{real}})$ by restricting attention to those terms in the summation with syndrome outcome $s = 0$. (See Appendix B for explanations of the notation for the logical Pauli $Q_{\ell(E)}$ induced by E and for the subset $\mathscr{E}_{\mathsf{X}}(Q) \subset \mathscr{E}$ of codes E for which $Q_{\ell(E)}$ is a purely Z-Pauli and Q has no X-error syndrome.) It is easy to see (and follows immediately from the work of Appendix B.2.1) that this unnormalized mixed state is

$$\underbrace{\frac{1}{|\mathscr{E}||\mathscr{P}||\mathscr{S}||\mathsf{B}||\tilde{\mathsf{B}}_{\mathsf{in}}|} \sum_{P,S,a,T^{\mathsf{in}},T^{\mathsf{out}},Q_{\tilde{\mathsf{M}}}} \sum_{E\in\mathscr{E}_{\mathsf{X}}(Q_{\tilde{\mathsf{M}}})} \langle a||Q_{\tilde{\mathsf{M}}}\rangle\langle Q_{\tilde{\mathsf{M}}}||a\rangle}_{\mathrm{Tr}_{\mathsf{K}}(R_{\text{acc}})} + \mathrm{Tr}_{\mathsf{K}}\left([\varepsilon_{\text{real}}]\right) \tag{35}$$

for some choice of $[\varepsilon_{\text{real}}]$. By the security of the decode-then-measure procedure used by the BR-OTP it must be that $\mathrm{Tr}_{\mathsf{K}}([\varepsilon_{\text{real}}])$ and hence $[\varepsilon_{\text{real}}]$ have trace at most ε. Henceforth we concentrate only on the highlighted term of (35), which is taken to be $\mathrm{Tr}_{\mathsf{K}}(R_{\text{acc}})$ for the operator R_{acc} appearing in the desired decomposition (32) of ρ_{real}.

Because the syndrome measurement succeeded, the BR-OTP reveals a new register K containing a classical description $|\hat{S}\rangle$ of a decryption key $\hat{S}$. (An explicit formula for this key is given in Section 6.6.3.) The unnormalized pure state $\langle a||Q_{\tilde{\mathsf{M}}}\rangle$ plus register K can be written

$$\langle a||Q_{\tilde{\mathsf{M}}}\rangle|\hat{S}\rangle = \sum_{\neg Q_{\tilde{\mathsf{M}}}} \alpha_{Q,T^{\mathsf{in}}}|(37)\rangle \tag{36}$$

where the table (37) is given by

W	Q_{W}					$\|\psi\rangle$
$\mathsf{B}_{\mathsf{out}}$	$Q_{\mathsf{B}_{\mathsf{out}}}E^*SQ_{\tilde{\mathsf{B}}_{\mathsf{out}}}T^{\mathsf{out}}Q_{\tilde{\mathsf{B}}_{\mathsf{in}}}$	$\pi_a(PT_E^{\mathsf{in}})_{\mathsf{B}_{\mathsf{out}}}$	E	$c\text{-}U_a$	$Q_{\mathsf{B}_{\mathsf{in}}}Q_{\mathsf{B}}$	
$\tilde{\mathsf{A}}$	$Q_{\tilde{\mathsf{A}}}$	$\pi_a(PT_E^{\mathsf{in}})_{\tilde{\mathsf{A}}}$	E			
$\tilde{\mathsf{E}}$	$Q_{\tilde{\mathsf{E}}}$	$\pi_a(PT_E^{\mathsf{in}})_{\tilde{\mathsf{E}}}$	E			$\|0\rangle\|\mathrm{on}\rangle$
$\tilde{\mathsf{M}}$			$\langle a\|$			$\|\mu\rangle$
K						$\|\hat{S}\rangle$

(37)

We may now employ the identity (4) from Section 6.1.1 to get

$$\langle a|c\text{-}U_a|\mu\rangle|\mathrm{on}\rangle = \frac{1}{2^{r/2}}c\text{-}U|\mathrm{on}\rangle = \frac{1}{2^{r/2}}U \tag{38}$$

so the state $|(37)\rangle$ becomes

$$|(37)\rangle = \frac{1}{2^{r/2}}$$

W	Q_{W}					$\|\psi\rangle$
$\mathsf{B}_{\mathsf{out}}$	$Q_{\mathsf{B}_{\mathsf{out}}}E^*SQ_{\tilde{\mathsf{B}}_{\mathsf{out}}}T^{\mathsf{out}}Q_{\tilde{\mathsf{B}}_{\mathsf{in}}}$	$\pi_a(PT_E^{\mathsf{in}})_{\mathsf{B}_{\mathsf{out}}}$	E	U	$Q_{\mathsf{B}_{\mathsf{in}}}Q_{\mathsf{B}}$	
$\tilde{\mathsf{A}}$	$Q_{\tilde{\mathsf{A}}}$	$\pi_a(PT_E^{\mathsf{in}})_{\tilde{\mathsf{A}}}$	E			
$\tilde{\mathsf{E}}$	$Q_{\tilde{\mathsf{E}}}$	$\pi_a(PT_E^{\mathsf{in}})_{\tilde{\mathsf{E}}}$	E			$\|0\rangle$
K						$\|\hat{S}\rangle$

(39)

Write

$$|Q_{\tilde{\mathsf{M}}}, \hat{S}\rangle \stackrel{\text{def}}{=} \sum_{\neg Q_{\tilde{\mathsf{M}}}} \alpha_{Q,T^{\mathsf{in}}}|(39)\rangle \tag{40}$$

so that

$$R_{\mathrm{acc}} = \frac{1}{|\mathscr{E}||\mathscr{P}||\mathscr{S}||\mathsf{B}||\tilde{\mathsf{B}}_{\mathsf{in}}|2^r} \sum_{P,S,a,T^{\mathsf{in}},T^{\mathsf{out}},Q_{\tilde{\mathsf{M}}}} \sum_{E\in\mathscr{E}_{\mathsf{X}}(Q_{\tilde{\mathsf{M}}})} |Q_{\tilde{\mathsf{M}}}, \hat{S}\rangle\langle Q_{\tilde{\mathsf{M}}}, \hat{S}| \tag{41}$$

Notice that the receiver/environment learns nothing about the keys $\pi_a(PT_E^{\mathsf{in}})_{\tilde{\mathsf{A}}}, \pi_a(PT_E^{\mathsf{in}})_{\tilde{\mathsf{E}}}$ so the registers $\tilde{\mathsf{A}}, \tilde{\mathsf{E}}$ remain completely mixed. Thus, the receiver gets only the $\mathsf{B}_{\mathsf{out}}$ portion of the final state as desired.

6.6.2 In the event of rejection

Consider the unnormalized mixed state obtained from the expression (31) for $\mathrm{Tr}_{\mathsf{K}}(\rho_{\text{real}})$ by restricting attention to those terms in the summation with syndrome outcome $s \neq 0$. (Again, the reader is reffered Appendix B for explanations of the notation $Q_{\ell(E)}$ and $\mathscr{E}_{\mathsf{X}\emptyset}(Q)$.) According to the analysis of Appendix B.2.1 this unnormalized mixed state is

$$\frac{1}{|\mathscr{E}||\mathscr{P}||\mathscr{S}||\mathsf{B}||\tilde{\mathsf{B}}_{\mathsf{in}}|} \sum_{P,S,a,s\neq 0,T^{\mathsf{in}},T^{\mathsf{out}},Q_{\tilde{\mathsf{M}}}} \sum_{E\in\mathscr{E}_{\mathsf{X}\emptyset}(Q_{\tilde{\mathsf{M}}})} \langle a|Q_{\tilde{\mathsf{M}},\ell(E)}|Q_{\tilde{\mathsf{M}}}\rangle\langle Q_{\tilde{\mathsf{M}}}|Q_{\tilde{\mathsf{M}},\ell(E)}|a\rangle \tag{42}$$

The operator of (42) shall be taken to be $\mathrm{Tr}_{\mathsf{K}}(R_{\text{rej}})$ for the operator R_{rej} appearing in the desired decomposition (32) of ρ_{real}.

Because the syndrome measurement failed, the register K revealed by the BR-OTP is completely mixed and so we can ignore it for the rest of this analysis. The unnormalized pure state $\langle a|Q_{\tilde{\mathsf{M}},\ell(E)}|Q_{\tilde{\mathsf{M}}}\rangle$ can be written

$$\langle a|Q_{\tilde{\mathsf{M}},\ell(E)}|Q_{\tilde{\mathsf{M}}}\rangle = \sum_{\neg Q_{\tilde{\mathsf{M}}}} \alpha_{Q,T^{\mathsf{in}}}|(44)\rangle \tag{43}$$

where the table (44) is given by

W		Q_{W}					$\|\psi\rangle$
$\mathsf{B}_{\mathsf{out}}$		$Q_{\mathsf{B}_{\mathsf{out}}}E^*SQ_{\tilde{\mathsf{B}}_{\mathsf{out}}}T^{\mathsf{out}}Q_{\tilde{\mathsf{B}}_{\mathsf{in}}}$	$\pi_a(PT_E^{\mathsf{in}})_{\mathsf{B}_{\mathsf{out}}}$	E	$c\text{-}U_a$	$Q_{\mathsf{B}_{\mathsf{in}}}Q_{\mathsf{B}}$	
$\tilde{\mathsf{A}}$		$Q_{\tilde{\mathsf{A}}}$	$\pi_a(PT_E^{\mathsf{in}})_{\tilde{\mathsf{A}}}$	E			
$\tilde{\mathsf{E}}$		$Q_{\tilde{\mathsf{E}}}$	$\pi_a(PT_E^{\mathsf{in}})_{\tilde{\mathsf{E}}}$	E			$\|0\rangle\|\text{on}\rangle$
$\tilde{\mathsf{M}}$	$\langle a\|$	$Q_{\tilde{\mathsf{M}},\ell(E)}$					$\|\mu\rangle$

(44)

For each $Q_{\tilde{\mathsf{M}},\ell(E)}$ let $|a'\rangle = Q_{\tilde{\mathsf{M}},\ell(E)}|a\rangle$ denote the modified vector of magic state measurement results. We may now employ the identity (4) from Section 6.1.1 to get

$$\langle a'|c\text{-}U_a|\mu\rangle = \frac{1}{2^{r/2}}c\text{-}U_{a-a'} \tag{45}$$

so the state $|(44)\rangle$ becomes

$$|(44)\rangle = \frac{1}{2^{r/2}}$$

W	Q_{W}					
$\mathsf{B}_{\mathsf{out}}$	$Q_{\mathsf{B}_{\mathsf{out}}}E^*SQ_{\tilde{\mathsf{B}}_{\mathsf{out}}}T^{\mathsf{out}}Q_{\tilde{\mathsf{B}}_{\mathsf{in}}}$	$\pi_a(PT_E^{\mathsf{in}})_{\mathsf{B}_{\mathsf{out}}}$	E	$c\text{-}U_{a-a'}$	$Q_{\mathsf{B}_{\mathsf{in}}}Q_{\mathsf{B}}$	$\|\psi\rangle$
$\tilde{\mathsf{A}}$	$Q_{\tilde{\mathsf{A}}}$	$\pi_a(PT_E^{\mathsf{in}})_{\tilde{\mathsf{A}}}$	E			
$\tilde{\mathsf{E}}$	$Q_{\tilde{\mathsf{E}}}$	$\pi_a(PT_E^{\mathsf{in}})_{\tilde{\mathsf{E}}}$	E			$\|0\rangle\|\text{on}\rangle$

(46)

In the case of rejection the sender has no information about the encryption Paulis $\pi_a(PT_E^{\mathsf{in}})_{\mathsf{B}_{\mathsf{out}}}$, $\pi_a(PT_E^{\mathsf{in}})_{\tilde{\mathsf{A}}}$, and $\pi_a(PT_E^{\mathsf{in}})_{\tilde{\mathsf{E}}}$. Thus, the unitary $Ec\text{-}U_{a-a'}$ appearing immediately before the encryption Paulis is superfluous—it can be removed without affecting the overall state. Moreover, the registers $\tilde{\mathsf{A}}, \tilde{\mathsf{E}}$ remain encrypted and therefore completely mixed from the receiver's point of view.

Thus, one could replace the state $|(44)\rangle$ with, say, the following

$$|(44)\rangle = \frac{1}{2^{r/2}} \begin{array}{|c||c|c|c|c|} \hline \mathsf{W} & Q_{\mathsf{W}} & & & |\psi\rangle_{\mathsf{W}} \\ \hline \mathsf{B_{out}} & Q_{\mathsf{B_{out}}} E^* S Q_{\tilde{\mathsf{B}}_{\mathsf{out}}} T^{\mathsf{out}} Q_{\tilde{\mathsf{B}}_{\mathsf{in}}} & \pi_a(PT_E^{\mathsf{in}})_{\mathsf{B_{out}}} & Q_{\mathsf{B_{in}}} Q_{\mathsf{B}} & |\psi\rangle_{\mathsf{B}} \\ \hline \tilde{\mathsf{A}} & Q_{\tilde{\mathsf{A}}} & \pi_a(PT_E^{\mathsf{in}})_{\tilde{\mathsf{A}}} & & |\text{anything}\rangle \\ \hline \tilde{\mathsf{E}} & Q_{\tilde{\mathsf{E}}} & \pi_a(PT_E^{\mathsf{in}})_{\tilde{\mathsf{E}}} & & |\text{anything}\rangle \\ \hline \end{array} \tag{47}$$

Letting

$$|Q_{\tilde{\mathsf{M}}}, \text{anything}\rangle \stackrel{\text{def}}{=} \sum_{\neg Q_{\tilde{\mathsf{M}}}} \alpha_{Q,T^{\mathsf{in}}} |(47)\rangle \tag{48}$$

we have

$$\mathrm{Tr}_{\mathsf{K}}(R_{\text{rej}}) = \frac{1}{|\mathscr{E}||\mathscr{P}||\mathscr{S}||\mathsf{B}||\tilde{\mathsf{B}}_{\mathsf{in}}|2^r} \sum_{P,S,a,s\neq 0,T^{\mathsf{in}},T^{\mathsf{out}},Q_{\tilde{\mathsf{M}}}} \sum_{E\in\mathscr{E}_{\mathsf{X}\emptyset}(Q_{\tilde{\mathsf{M}}})} |Q_{\tilde{\mathsf{M}}}, \text{anything}\rangle\langle Q_{\tilde{\mathsf{M}}}, \text{anything}| \tag{49}$$

6.6.3 More analysis to determine the final key

Now that we have analyzed an arbitrary environment/receiver for our QOTP we can easily derive an expression for the decryption key $\hat{S}$ used by an honest receiver to recover his output register B at the end of the protocol. The derivation in this section is not a prerequisite for the discussion of the simulator in Section 6.7. It is included here only for completeness.

We claim that the decryption Pauli $\hat{S}$ is the logical Pauli induced on the data register by code E and Pauli

$$S T^{\mathsf{out}} \pi_a(PT_E^{\mathsf{in}})_{\mathsf{B_{out}}} \quad . \tag{50}$$

To see this, observe that an honest receiver will place no amplitude on terms for which $Q \neq I$. Our calculations are therefore simplified by assuming $Q = I$. Under this assumption the state $|(39)\rangle$ becomes

$$|(39)\rangle = \begin{array}{|c||c|c|c|c|} \hline \mathsf{W} & & & & \\ \hline \mathsf{B_{out}} & E^* S T^{\mathsf{out}} & \pi_a(PT_E^{\mathsf{in}})_{\mathsf{B_{out}}} & E & U \quad |\psi\rangle \\ \hline \tilde{\mathsf{A}} & & \pi_a(PT_E^{\mathsf{in}})_{\tilde{\mathsf{A}}} & E & \\ \hline \tilde{\mathsf{E}} & & \pi_a(PT_E^{\mathsf{in}})_{\tilde{\mathsf{E}}} & E & |0\rangle \\ \hline \mathsf{K} & & & & |\hat{S}\rangle \\ \hline \end{array} \tag{51}$$

and so we see that applying $\hat{S}$ to register $\mathsf{B_{out}}$ decrypts that register. Of course, $\tilde{\mathsf{A}}, \tilde{\mathsf{E}}$ remain encrypted and the purification register W is not in the receiver's posession, so the receiver obtains only the B portion of $(\Phi \otimes \mathbb{1}_{\mathsf{W}})(|\psi\rangle\langle\psi|)$ as desired.

6.7 Specification of the simulator

In Section 6.6 we derived an expression for the mixed state ρ_{real} of the registers $(\mathsf{B_{out}}, \tilde{\mathsf{A}}, \tilde{\mathsf{E}}, \mathsf{W}, \mathsf{K})$ in the environment's possession at the end of step 5. In this section we specify a simulator, and

in the following section we show that the state ρ_{sim} of the environment's registers at the end of its interaction with the simulator is close in trace distance to ρ_{real}, from which security is established.

This simulator must not interact with the sender. Instead, the simulator is permitted only one-shot, black-box access to the "ideal functionality" for Φ. We represent this ideal functionality by a single call to an oracle for U acting on registers $(\mathsf{A}, \mathsf{B}, \mathsf{E})$ prepared by the simulator. The rules for permissible preparation and disposal of these registers are as follows:

1. When given the initial state $|\psi\rangle$ of registers $(\mathsf{A}, \mathsf{B}, \mathsf{W})$ selected by the environment, the simulator *must* pass the register A directly to U without any pre-processing.
2. The simulator must prepare the ancillary register E in pure state $|0\rangle$.
3. Upon receiving the output registers $(\mathsf{A}, \mathsf{B}, \mathsf{E})$, the simulator must discard registers A, E without any post-processing.

The simulator constructs registers as listed in Protocol 5 and then acts as specified described in the protocol. The main idea is that our simulator will use the control bit contained in register $\tilde{\mathsf{E}}$ to "switch off" the application of U that would have been implemented by the real receiver interacting with the sender's BR-OTP. Instead, the black-box call to the ideal functionality will be embedded at the proper time so as to recover the required action of U. An additional teleportation step is required so that our simulator can embed U at the proper time.

Protocol 5 Simulator

Registers prepared by the simulator.

Given the input register A our simulator constructs the following registers:

$(\mathsf{B}_{\mathsf{in}}, \mathsf{S}_{\mathsf{in}})$:	Simple EPR pairs $\|\phi^+\rangle$ for teleportation.
$(\mathsf{S}_{\mathsf{out}}, \tilde{\mathsf{B}}_{\mathsf{in}})$:	Teleport-through-authentication state of Protocol 1.
$(\tilde{\mathsf{B}}_{\mathsf{out}}, \mathsf{B}_{\mathsf{out}})$:	Teleport-through-de-authentication state of Protocol 1.
$\tilde{\mathsf{A}}$:	Authenticated dummy input state for the sender $P_{\tilde{\mathsf{A}}}E\|0\rangle$.
$\tilde{\mathsf{E}}$:	Authenticated dummy ancillary state $P_{\tilde{\mathsf{E}}}E\|0\rangle\|\text{off}\rangle$.
$\tilde{\mathsf{M}}$:	Authenticated magic states as in Protocol 1.
A' :	To be used in the call to the ideal functionality. Contains the portion of $\|\psi\rangle$ contained in register A.
E' :	To be used in the call to the ideal functionality. Ancillary register in state $\|0\rangle$.

Execution of the simulator.

1. Prepare registers $(\mathsf{B}_{\mathsf{in}}, \tilde{\mathsf{B}}_{\mathsf{in}}, \tilde{\mathsf{B}}_{\mathsf{out}}, \mathsf{B}_{\mathsf{out}}, \tilde{\mathsf{A}}, \tilde{\mathsf{E}}, \tilde{\mathsf{M}}, \mathsf{W})$ as above and send these registers to the environment.
2. The environment responds with a Pauli T^{in}. Apply T^{in} to register S_{in}. Then use the ideal black-box to apply U to $(\mathsf{A}', \mathsf{S}_{\mathsf{in}}, \mathsf{E}')$.
3. Perform a Bell measurement on $(\mathsf{S}_{\mathsf{in}}, \mathsf{S}_{\mathsf{out}})$ so as to teleport the contents of S_{in} through the authentication and place the result in $\tilde{\mathsf{B}}_{\mathsf{in}}$. Let T^{sim} denote the teleportation Pauli indicated by this Bell measurement.
4. Execute Protocol 2 for the BR-OTP under the assumption that T^{sim} was received in the first round. (This step is not difficult: the responses of the BR-OTP depend only upon the choice of code E and encryption Pauli P.)

6.8 Analysis of the environment's interaction with the simulator

Fix a choice of code E and encryption Paulis P, S. We re-use notation from Section 6.6 as much as possible. In Section 6.5 we wrote the unnormalized pure state of the entire system as a large table $|(6)\rangle$. We then introduced some shorthand notation in Section 6.6 that allowed us to write that table more compactly as $|(11)\rangle$. The equivalent of table (11) for our simulator is as follows.

E'			$(U)_{\mathsf{E}}$	$\|0\rangle$
A'			$(U)_{\mathsf{A}}$	$\|\psi\rangle_{\mathsf{A}}$
B	$\langle T^{\mathsf{in}}\|B$	$(K_a)_{\mathsf{B}}$		$\|\psi\rangle_{\mathsf{B}}$
B_{in}		$(K_a)_{\mathsf{B}_{\mathsf{in}}}$		$\|\phi^+\rangle$
S_{in}	$\langle T^{\mathsf{sim}}\|B$		$(U)_{\mathsf{B}}T^{\mathsf{in}}$	
$\mathsf{S}_{\mathsf{out}}$				$\|\phi^+\rangle$
$\tilde{\mathsf{B}}_{\mathsf{in}}$	$\langle T^{\mathsf{out}}\|B$		$P_{\tilde{\mathsf{B}}_{\mathsf{in}}}E$	
$\tilde{\mathsf{B}}_{\mathsf{out}}$				$\|\phi^+\rangle$
$\mathsf{B}_{\mathsf{out}}$			E^*S	
$\tilde{\mathsf{A}}$		K_a	$P_{\tilde{\mathsf{A}}}E$	$\|0\rangle$
$\tilde{\mathsf{E}}$			$P_{\tilde{\mathsf{E}}}E$	$\|0\rangle\|\text{off}\rangle$
$\tilde{\mathsf{M}}$	$\langle a\|\langle s\|E^*\hat{P}_{\tilde{\mathsf{M}}}$		$P_{\tilde{\mathsf{M}}}E$	$\|\mu\rangle$
W				$\|\psi\rangle_{\mathsf{W}}$

(52)

Then

$$\mathrm{Tr}_{\mathsf{K}}(\rho_{\mathrm{sim}}) = \frac{1}{|\mathscr{E}||\mathscr{P}||\mathscr{S}|} \sum_{E,P,S,a,s,T^{\mathsf{in}},T^{\mathsf{out}},T^{\mathsf{sim}}} \mathrm{Tr}_{\mathsf{A}'\mathsf{E}'}\left(|(52)\rangle\langle(52)|\right) \quad . \tag{53}$$

We then inserted a superfluous $I = (c\text{-}\tilde{U}_a)^*(c\text{-}\tilde{U}_a)$ into the table (11) and derived the table (15) by substituting the Pauli decomposition (13) of $K_a(c\text{-}\tilde{U}_a)^*$ and fixing a choice Q of Pauli in that decomposition. It is a simple matter to repeat those steps in the current setting. The equivalent

of table (15) for our simulator is

E'				$(U)_{\mathsf{E}'}$	$\|0\rangle$
A'				$(U)_{\mathsf{A}'}$	$\|\psi\rangle_{\mathsf{A}}$
B	$\langle T^{\mathsf{in}}\|B$	Q_{B}			$\|\psi\rangle_{\mathsf{B}}$
B_{in}		$Q_{\mathsf{B}_{\mathsf{in}}}$			$\|\phi^+\rangle$
S_{in}	$\langle T^{\mathsf{sim}}\|B$			$(U)_{\mathsf{B}}T^{\mathsf{in}}$	
$\mathsf{S}_{\mathsf{out}}$					$\|\phi^+\rangle$
$\tilde{\mathsf{B}}_{\mathsf{in}}$	$\langle T^{\mathsf{out}}\|B$	$Q_{\tilde{\mathsf{B}}_{\mathsf{in}}}$	$(c\text{-}\tilde{U}_a)_{\tilde{\mathsf{B}}_{\mathsf{in}}}$	$P_{\tilde{\mathsf{B}}_{\mathsf{in}}}E$	
$\tilde{\mathsf{B}}_{\mathsf{out}}$		$Q_{\tilde{\mathsf{B}}_{\mathsf{out}}}$			$\|\phi^+\rangle$
$\mathsf{B}_{\mathsf{out}}$		$Q_{\mathsf{B}_{\mathsf{out}}}$		E^*S	
$\tilde{\mathsf{A}}$		$Q_{\tilde{\mathsf{A}}}$	$(c\text{-}\tilde{U}_a)_{\tilde{\mathsf{A}}}$	$P_{\tilde{\mathsf{A}}}E$	$\|0\rangle$
$\tilde{\mathsf{E}}$		$Q_{\tilde{\mathsf{E}}}$	$(c\text{-}\tilde{U}_a)_{\tilde{\mathsf{E}}}$	$P_{\tilde{\mathsf{E}}}E$	$\|0\rangle\|\text{off}\rangle$
$\tilde{\mathsf{M}}$	$\langle a\|\langle s\|E^*\hat{P}'_{\tilde{\mathsf{M}}}$	$Q_{\tilde{\mathsf{M}}}$	$(c\text{-}\tilde{U}_a)_{\tilde{\mathsf{M}}}$	$P_{\tilde{\mathsf{M}}}E$	$\|\mu\rangle$
W		Q_{W}			$\|\psi\rangle_{\mathsf{W}}$

(54)

In Section 6.6 we applied a teleportation identity to derive table (18) from table (15). We apply the same teleportation identity here so as to derive the following table from (54).

E'						U		$\|0\rangle$
A'								$\|\psi\rangle_{\mathsf{A}}$
$\mathsf{B}_{\mathsf{out}}$		$Q_{\mathsf{B}_{\mathsf{out}}}E^*SQ^{\mathsf{T}}_{\tilde{\mathsf{B}}_{\mathsf{out}}}T^{\mathsf{out}}Q_{\tilde{\mathsf{B}}_{\mathsf{in}}}$			ET^{sim}		$T^{\mathsf{in}}Q^{\mathsf{T}}_{\mathsf{B}_{\mathsf{in}}}T^{\mathsf{in}}Q_{\mathsf{B}}$	$\|\psi\rangle_{\mathsf{B}}$
$\tilde{\mathsf{A}}$		$Q_{\tilde{\mathsf{A}}}$	$c\text{-}\tilde{U}_a$	P	E			$\|0\rangle$
$\tilde{\mathsf{E}}$		$Q_{\tilde{\mathsf{E}}}$			E			$\|0\rangle\|\text{off}\rangle$
$\tilde{\mathsf{M}}$	$\langle a\|\langle s\|E^*\hat{P}'_{\tilde{\mathsf{M}}}$	$Q_{\tilde{\mathsf{M}}}$			E			$\|\mu\rangle$
W		Q_{W}						$\|\psi\rangle_{\mathsf{W}}$

(55)

We then replaced transposed Paulis with their un-transposed equivalents and employed several commutation relations to derive table (25). That derivation can be repeated almost exactly in the present context. The only significant difference is that, after commuting $Q_{\mathsf{B}_{\mathsf{in}}}$ with T^{in} and picking up a phase of $c(T^{\mathsf{in}}, Q_{\mathsf{B}_{\mathsf{in}}})$, the two T^{in} Paulis annihilate each other. The state $|(54)\rangle$ becomes

E'						U		$\|0\rangle$
A'								$\|\psi\rangle_{\mathsf{A}}$
$\mathsf{B}_{\mathsf{out}}$		$Q_{\mathsf{B}_{\mathsf{out}}}E^*SQ_{\tilde{\mathsf{B}}_{\mathsf{out}}}T^{\mathsf{out}}Q_{\tilde{\mathsf{B}}_{\mathsf{in}}}$		E			$Q_{\mathsf{B}_{\mathsf{in}}}Q_{\mathsf{B}}$	$\|\psi\rangle_{\mathsf{B}}$
$\tilde{\mathsf{A}}$		$Q_{\tilde{\mathsf{A}}}$	$\pi_a(PT^{\mathsf{sim}}_E)$	E	$c\text{-}U_a$			$\|0\rangle$
$\tilde{\mathsf{E}}$		$Q_{\tilde{\mathsf{E}}}$		E				$\|0\rangle\|\text{off}\rangle$
$\tilde{\mathsf{M}}$	$\langle a\|\langle s\|E^*\hat{P}'_{\tilde{\mathsf{M}}}$	$Q_{\tilde{\mathsf{M}}}$		E				$\|\mu\rangle$
W		Q_{W}						$\|\psi\rangle_{\mathsf{W}}$

(56)

We now see that the decryption Pauli $\hat{P}'_{\tilde{\mathsf{M}}}$ used in the BR-OTP is given by

$$\hat{P}'_{\tilde{\mathsf{M}}} \stackrel{\text{def}}{=} \pi_a(PT_E^{\mathsf{sim}})_{\tilde{\mathsf{M}}} \ . \tag{57}$$

By analogy to the formula (28) for the state $|(6)\rangle$ in Section 6.6 we have

$$|(52)\rangle = \Big(|\mathsf{B}||\tilde{\mathsf{B}}_{\mathsf{in}}||\mathsf{S}_{\mathsf{in}}|\Big)^{-1/2} \sum_{\text{Paulis } Q} \alpha_{Q,T^{\mathsf{in}}} |(56)\rangle \ . \tag{58}$$

As before, we wish to employ the Pauli sandwich on $\tilde{\mathsf{M}}$. To this end consider the table

E'					U		$\|0\rangle$
A'							$\|\psi\rangle_{\mathsf{A}}$
$\mathsf{B}_{\mathsf{out}}$	$Q_{\mathsf{B}_{\mathsf{out}}} E^* S Q_{\tilde{\mathsf{B}}_{\mathsf{out}}} T^{\mathsf{out}} Q_{\tilde{\mathsf{B}}_{\mathsf{in}}}$	$\pi_a(PT_E^{\mathsf{sim}})_{\mathsf{B}_{\mathsf{out}}}$	E	$c\text{-}U_a$		$Q_{\mathsf{B}_{\mathsf{in}}} Q_{\mathsf{B}}$	$\|\psi\rangle_{\mathsf{B}}$
$\tilde{\mathsf{A}}$	$Q_{\tilde{\mathsf{A}}}$	$\pi_a(PT_E^{\mathsf{sim}})_{\tilde{\mathsf{A}}}$	E				$\|0\rangle$
$\tilde{\mathsf{E}}$	$Q_{\tilde{\mathsf{E}}}$	$\pi_a(PT_E^{\mathsf{sim}})_{\tilde{\mathsf{E}}}$	E				$\|0\rangle\|\text{off}\rangle$
$\tilde{\mathsf{M}}$							$\|\mu\rangle$
W	Q_{W}						$\|\psi\rangle_{\mathsf{W}}$

(59)

obtained from (56) by deleting everything in row $\tilde{\mathsf{M}}$ occurring after the operator $c\text{-}U_a$. By analogy to the state $|Q_{\tilde{\mathsf{M}}}\rangle$ defined in (30) in Section 6.6, for each Pauli $Q_{\tilde{\mathsf{M}}}$ define

$$|Q'_{\tilde{\mathsf{M}}}\rangle \stackrel{\text{def}}{=} \sum_{\neg Q_{\tilde{\mathsf{M}}}} \alpha_{Q,T_{\mathsf{in}}} |(59)\rangle \ . \tag{60}$$

By analogy to the expression (31) for $\mathrm{Tr}_{\mathsf{K}}(\rho_{\text{real}})$ derived in Section 6.6, by the Pauli sandwich (Lemma 13) we have that the mixed state $\mathrm{Tr}_{\mathsf{K}}(\rho_{\text{sim}})$ from (53) equals

$$\mathrm{Tr}_{\mathsf{K}}(\rho_{\text{sim}}) = \frac{1}{|\mathscr{E}||\mathscr{P}||\mathscr{S}||\mathsf{B}||\tilde{\mathsf{B}}_{\mathsf{in}}||\mathsf{S}_{\mathsf{in}}|} \sum_{E,P,S,a,s,T^{\mathsf{in}},T^{\mathsf{out}},T^{\mathsf{sim}},Q_{\tilde{\mathsf{M}}}} \langle a|\langle s|E^* Q_{\tilde{\mathsf{M}}} E \,\mathrm{Tr}_{\mathsf{A}'\mathsf{E}'} \left(|Q'_{\tilde{\mathsf{M}}}\rangle\langle Q'_{\tilde{\mathsf{M}}}|\right) E^* Q_{\tilde{\mathsf{M}}} E |a\rangle|s\rangle \ . \tag{61}$$

As mentioned in Section 6.6, we will derive the expression (33) for ρ_{sim}, from which it will follow that the state is close to ρ_{real} in trace distance as desired.

6.8.1 In the event of acceptance

Just as in Section 6.6.1, consider the unnormalized mixed state obtained from the expression (61) for $\mathrm{Tr}_{\mathsf{K}}(\rho_{\text{sim}})$ by restricting attention to those terms in the summation with syndrome outcome $s = 0$. We can follow the argument of Section 6.6.1 to see that this unnormalized mixed state is

$$\underbrace{\frac{1}{|\mathscr{E}||\mathscr{P}||\mathscr{S}||\mathsf{B}||\tilde{\mathsf{B}}_{\mathsf{in}}||\mathsf{S}_{\mathsf{in}}|} \sum_{P,S,a,T^{\mathsf{in}},T^{\mathsf{out}},T^{\mathsf{sim}},Q_{\tilde{\mathsf{M}}}} \sum_{E \in \mathscr{E}_{\mathsf{X}}(Q_{\tilde{\mathsf{M}}})} \langle a| \,\mathrm{Tr}_{\mathsf{A}'\mathsf{E}'} \left(|Q'_{\tilde{\mathsf{M}}}\rangle\langle Q'_{\tilde{\mathsf{M}}}|\right) |a\rangle}_{\text{show equal to } \mathrm{Tr}_{\mathsf{K}}(R_{\text{acc}})} + \mathrm{Tr}_{\mathsf{K}}\left([\varepsilon_{\text{sim}}]\right) \tag{62}$$

for some choice of $[\varepsilon_{\text{sim}}]$ with trace at most ε. Henceforth we concentrate only on the indicated term of (62). We will show that this term equals $\text{Tr}_{\mathsf{K}}(R_{\text{acc}})$.

Because the syndrome measurement succeeded, our simulator reveals a new register K containing a classical description $|\hat{S}'\rangle$ of a decryption key $\hat{S}'$. (More about this key later in this section.) The unnormalized pure state $\langle a||Q'_{\tilde{\mathsf{M}}}\rangle$ plus register K can be written

$$\langle a||Q'_{\tilde{\mathsf{M}}}\rangle|\hat{S}'\rangle = \sum_{\neg Q_{\tilde{\mathsf{M}}}} \alpha_{Q,T^{\mathsf{in}}}|(64)\rangle \tag{63}$$

where the table (64) is given by

E'					U		$\|0\rangle$
A'							$\|\psi\rangle_{\mathsf{A}}$
$\mathsf{B}_{\mathsf{out}}$	$Q_{\mathsf{B}_{\mathsf{out}}} E^* S Q_{\tilde{\mathsf{B}}_{\mathsf{out}}} T^{\mathsf{out}} Q_{\tilde{\mathsf{B}}_{\mathsf{in}}}$	$\pi_a(PT_E^{\mathsf{sim}})_{\mathsf{B}_{\mathsf{out}}}$	E	$c\text{-}U_a$		$Q_{\mathsf{B}_{\mathsf{in}}} Q_{\mathsf{B}}$	$\|\psi\rangle_{\mathsf{B}}$
$\tilde{\mathsf{A}}$	$Q_{\tilde{\mathsf{A}}}$	$\pi_a(PT_E^{\mathsf{sim}})_{\tilde{\mathsf{A}}}$	E				$\|0\rangle$
$\tilde{\mathsf{E}}$	$Q_{\tilde{\mathsf{E}}}$	$\pi_a(PT_E^{\mathsf{sim}})_{\tilde{\mathsf{E}}}$	E				$\|0\rangle\|\text{off}\rangle$
$\tilde{\mathsf{M}}$			$\langle a\|$				$\|\mu\rangle$
W	Q_{W}						$\|\psi\rangle_{\mathsf{W}}$
K							$\|\hat{S}'\rangle$

(64)

Similar to Section 6.6, (except now the control-bit switch is set to $|\text{off}\rangle$) we may now employ the identity (4) from Section 6.1.1 to get

$$\langle a|c\text{-}U_a|\mu\rangle|\text{off}\rangle = \frac{1}{2^{r/2}} c\text{-}U|\text{off}\rangle = \frac{1}{2^{r/2}} I \tag{65}$$

so the state $|(64)\rangle$ becomes

$$|(64)\rangle = \frac{1}{2^{r/2}}$$

W	Q_{W}					$\|\psi\rangle_{\mathsf{W}}$
E'				U		$\|0\rangle$
A'						$\|\psi\rangle_{\mathsf{A}}$
$\mathsf{B}_{\mathsf{out}}$	$Q_{\mathsf{B}_{\mathsf{out}}} E^* S Q_{\tilde{\mathsf{B}}_{\mathsf{out}}} T^{\mathsf{out}} Q_{\tilde{\mathsf{B}}_{\mathsf{in}}}$	$\pi_a(PT_E^{\mathsf{in}})_{\mathsf{B}_{\mathsf{out}}}$	E		$Q_{\mathsf{B}_{\mathsf{in}}} Q_{\mathsf{B}}$	$\|\psi\rangle_{\mathsf{B}}$
$\tilde{\mathsf{A}}$	$Q_{\tilde{\mathsf{A}}}$	$\pi_a(PT_E^{\mathsf{in}})_{\tilde{\mathsf{A}}}$	E			$\|0\rangle$
$\tilde{\mathsf{E}}$	$Q_{\tilde{\mathsf{E}}}$	$\pi_a(PT_E^{\mathsf{in}})_{\tilde{\mathsf{E}}}$	E			$\|0\rangle$
K						$\|\hat{S}'\rangle$

(66)

Write

$$|Q'_{\tilde{\mathsf{M}}}, \hat{S}'\rangle \stackrel{\text{def}}{=} \sum_{\neg Q_{\tilde{\mathsf{M}}}} \alpha_{Q,T^{\mathsf{in}}}|(66)\rangle \tag{67}$$

so that the highlighted term in (62) becomes

$$\frac{1}{|\mathscr{E}||\mathscr{P}||\mathscr{S}||\mathsf{B}||\tilde{\mathsf{B}}_{\mathsf{in}}||\mathsf{S}_{\mathsf{in}}|2^r} \sum_{P,S,a,T^{\mathsf{in}},T^{\mathsf{out}},T^{\mathsf{sim}},Q_{\tilde{\mathsf{M}}}} \sum_{E\in\mathscr{E}_{\mathsf{X}}(Q_{\tilde{\mathsf{M}}})} \text{Tr}_{\mathsf{A}'\mathsf{E}'}\left(|Q'_{\tilde{\mathsf{M}}}, \hat{S}'\rangle\langle Q'_{\tilde{\mathsf{M}}}, \hat{S}'|\right) \quad . \tag{68}$$

By repeating the analysis of Section 6.6.3 it is easy to see that the final decryption Pauli $\hat{S}'$ produced by the simulator is the logical Pauli induced on the data register by code E and Pauli

$$ST^{\mathsf{out}}\pi_a(PT_E^{\mathsf{sim}})_{\tilde{\mathsf{B}}_{\mathsf{out}}} \ . \tag{69}$$

Finally, we claim that the expression (68) equals R_{acc} from (41). To justify this claim we observe the differences between the tables (39) and (66). The only differences are

1. T_E^{in} has been replaced with T_E^{sim} in the encryption Paulis for $\mathsf{B}_{\mathsf{out}}, \tilde{\mathsf{A}}, \tilde{\mathsf{E}}$ and in the final decryption key $\hat{S}'$.
2. U is applied to the auxiliary registers A', E' that are discarded by the simulator instead of the registers $\tilde{\mathsf{A}}, \tilde{\mathsf{E}}$ in the environment's possession. The sender's portion of the input state $|\psi\rangle_{\mathsf{A}}$ is contained in register A' instead of $\tilde{\mathsf{A}}$.

The first difference is eliminated by the uniformly random encryption Pauli P; the summation over T^{sim} in (68) can be eliminated by a simple change of variable, summing over $PT_E^{\mathsf{in}}T_E^{\mathsf{sim}}$ instead of P. The second difference is trivial because the registers $\tilde{\mathsf{A}}, \tilde{\mathsf{E}}$ are encrypted anyway. In particular, $|(39)\rangle$ can be obtained from $|(66)\rangle$ by swapping $(\mathsf{A}', \mathsf{E}')$ for $(\tilde{\mathsf{A}}, \tilde{\mathsf{E}})$ prior to encryption of those registers. Since $\tilde{\mathsf{A}}, \tilde{\mathsf{E}}$ are encrypted, such a swap cannot be detected.

6.8.2 In the event of rejection

Just as in Section 6.6.2, consider the unnormalized mixed state obtained from the expression (61) for $\text{Tr}_{\mathsf{K}}(\rho_{\text{sim}})$ by restricting attention to those terms in the summation with syndrome outcome $s \neq 0$. We can follow the argument of Section 6.6.2 to see that this unnormalized mixed state is

$$\frac{1}{|\mathscr{E}||\mathscr{P}||\mathscr{S}||\mathsf{B}||\tilde{\mathsf{B}}_{\mathsf{in}}||\mathsf{S}_{\mathsf{in}}|} \sum_{P,S,a,s\neq 0,T^{\mathsf{in}},T^{\mathsf{out}},T^{\mathsf{sim}},Q_{\tilde{\mathsf{M}}}} \sum_{E\in\mathscr{E}_{\mathsf{X}\emptyset}(Q_{\tilde{\mathsf{M}}})} \langle a|Q_{\tilde{\mathsf{M}},\ell(E)} \, \text{Tr}_{\mathsf{A}'\mathsf{E}'}\left(|Q'_{\tilde{\mathsf{M}}}\rangle\langle Q'_{\tilde{\mathsf{M}}}|\right) Q^*_{\tilde{\mathsf{M}},\ell(E)}|a\rangle \ . \tag{70}$$

We will show that the expression (70) equals $\text{Tr}_{\mathsf{K}}(R_{\text{rej}})$.

Because the syndrome measurement failed, the register K revealed by the simulator is completely mixed and so we can ignore it for the rest of this analysis. The unnormalized pure state $\langle a|Q_{\tilde{\mathsf{M}},\ell(E)}|Q'_{\tilde{\mathsf{M}}}\rangle$ can be written

$$\langle a|Q_{\tilde{\mathsf{M}},\ell(E)}|Q'_{\tilde{\mathsf{M}}}\rangle = \sum_{\neg Q_{\tilde{\mathsf{M}}}} \alpha_{Q,T^{\mathsf{in}}}|(72)\rangle \tag{71}$$

where the table (72) is given by

E'						U		$\|0\rangle$
A'								$\|\psi\rangle_{\mathsf{A}}$
$\mathsf{B}_{\mathsf{out}}$		$Q_{\mathsf{B}_{\mathsf{out}}}E^*SQ_{\tilde{\mathsf{B}}_{\mathsf{out}}}T^{\mathsf{out}}Q_{\tilde{\mathsf{B}}_{\mathsf{in}}}$	$\pi_a(PT_E^{\mathsf{sim}})_{\mathsf{B}_{\mathsf{out}}}$	E	$c\text{-}U_a$		$Q_{\mathsf{B}_{\mathsf{in}}}Q_{\mathsf{B}}$	$\|\psi\rangle_{\mathsf{B}}$
$\tilde{\mathsf{A}}$		$Q_{\tilde{\mathsf{A}}}$	$\pi_a(PT_E^{\mathsf{sim}})_{\tilde{\mathsf{A}}}$	E				$\|0\rangle$
$\tilde{\mathsf{E}}$		$Q_{\tilde{\mathsf{E}}}$	$\pi_a(PT_E^{\mathsf{sim}})_{\tilde{\mathsf{E}}}$	E				$\|0\rangle\|\text{off}\rangle$
$\tilde{\mathsf{M}}$	$\langle a\|$	$Q_{\tilde{\mathsf{M}}}$						$\|\mu\rangle$
W		Q_{W}						$\|\psi\rangle_{\mathsf{W}}$

(72)

For each $Q_{\tilde{\mathsf{M}},\ell(E)}$ let $|a'\rangle = Q_{\tilde{\mathsf{M}},\ell(E)}|a\rangle$ denote the modified vector of magic state measurement results. As in Section 6.6.2 we may now employ the identity (4) from Section 6.1.1 to get

$$\langle a'|c\text{-}U_a|\mu\rangle = \frac{1}{2^{r/2}} c\text{-}U_{a-a'} \tag{73}$$

so the state $|(72)\rangle$ becomes

$|(72)\rangle = \frac{1}{2^{r/2}}$

W	Q_{W}						$\|\psi\rangle_{\mathsf{W}}$
E'							$\|0\rangle$
A'					U		$\|\psi\rangle_{\mathsf{A}}$
$\mathsf{B}_{\mathsf{out}}$	$Q_{\mathsf{B}_{\mathsf{out}}} E^* S Q_{\tilde{\mathsf{B}}_{\mathsf{out}}} T^{\mathsf{out}} Q_{\tilde{\mathsf{B}}_{\mathsf{in}}}$	$\pi_a(PT_E^{\mathsf{sim}})_{\mathsf{B}_{\mathsf{out}}}$	E			$Q_{\mathsf{B}_{\mathsf{in}}} Q_{\mathsf{B}}$	$\|\psi\rangle_{\mathsf{B}}$
$\tilde{\mathsf{A}}$	$Q_{\tilde{\mathsf{A}}}$	$\pi_a(PT_E^{\mathsf{sim}})_{\tilde{\mathsf{A}}}$	E	$c\text{-}U_{a-a'}$			$\|0\rangle$
$\tilde{\mathsf{E}}$	$Q_{\tilde{\mathsf{E}}}$	$\pi_a(PT_E^{\mathsf{sim}})_{\tilde{\mathsf{E}}}$	E				$\|0\rangle\|\text{off}\rangle$

(74)

In the case of rejection the sender has no information about the encryption Paulis $\pi_a(PT_E^{\mathsf{in}})_{\mathsf{B}_{\mathsf{out}}}$, $\pi_a(PT_E^{\mathsf{in}})_{\tilde{\mathsf{A}}}$, and $\pi_a(PT_E^{\mathsf{in}})_{\tilde{\mathsf{E}}}$. Thus, any unitaries that appear immediately before the encryption Paulis are superfluous—they can be removed without affecting the overall state. In particular, we get the same state even after we replace $Ec\text{-}U_{a-a'}$ with the identity in the table (74). Furthermore, since A', E' are discarded by the simulator, the same logic allows us to replace U with the identity. Finally, since registers $(\tilde{\mathsf{A}}, \tilde{\mathsf{E}})$ remain encrypted, one could replace the state $|(74)\rangle$ with

$|(72)\rangle = \frac{1}{2^{r/2}}$

E'				$\|0\rangle$
A'				$\|\psi\rangle_{\mathsf{A}}$
$\mathsf{B}_{\mathsf{out}}$	$Q_{\mathsf{B}_{\mathsf{out}}} E^* S Q_{\tilde{\mathsf{B}}_{\mathsf{out}}} T^{\mathsf{out}} Q_{\tilde{\mathsf{B}}_{\mathsf{in}}}$	$\pi_a(PT_E^{\mathsf{sim}})_{\mathsf{B}_{\mathsf{out}}}$	$Q_{\mathsf{B}_{\mathsf{in}}} Q_{\mathsf{B}}$	$\|\psi\rangle_{\mathsf{B}}$
$\tilde{\mathsf{A}}$	$Q_{\tilde{\mathsf{A}}}$	$\pi_a(PT_E^{\mathsf{sim}})_{\tilde{\mathsf{A}}}$		$\|\text{anything}\rangle$
$\tilde{\mathsf{E}}$	$Q_{\tilde{\mathsf{E}}}$	$\pi_a(PT_E^{\mathsf{sim}})_{\tilde{\mathsf{E}}}$		$\|\text{anything}\rangle$
W	Q_{W}			$\|\psi\rangle_{\mathsf{W}}$

(75)

Summing over $PT_E^{\mathsf{in}} T_E^{\mathsf{sim}}$ instead of P and discarding registers $(\mathsf{A}', \mathsf{E}')$ we see that the above state can be interchanged with $|(47)\rangle$ from Section 6.6.2. The desired expression for R_{rej} follows.

6.9 Result

We have thus shown that, no environment of the given in Section 6.4 can distinguish ρ_{real}, as calculated in Section 6.6, from the output ρ_{sim}, as calculated in Section 6.8. Moreover, we argued in Section 6.4 that any arbitrary environment is equivalent to an environment of the stated form.

This yields the proof of Theorem 8, that the protocol described in Sections 6.2 and 6.3 is statistically quantum-UC-secure realization of $\mathcal{F}_{\Phi}^{\text{OTP}}$ in the case of a corrupt user, in the $\mathcal{F}^{\text{BR-OTP}}$-hybrid model.

By combining this with the result that $\mathcal{F}^{\text{BR-OTP}}$ can be realized statistically UC-secure in the $\mathcal{F}^{\text{OTM}}$-hybrid model (Corollary 2.1), the quantum lifting theorem [Unr10], and the quantum UC transitivity lemma (Lemma 1), we achieve the central result of the paper, Theorem 7: a

protocol for non-interactive statistically quantum-UC-secure (in the case of an honest sender and potentially corrupt receiver) one-time programs, assuming secure OTM tokens (i.e., in the $\mathcal{F}^{\mathrm{OTM}}$-hybrid model). □

7 UC security of delegating quantum computations

We show in this section that our main proof technique can also be used to establish the statistical quantum-UC-security of a family of protocols for delegating quantum computations, closely related to the protocol of Aharonov *et al.* [ABOE10]. Originally studied in the context of *quantum interactive proof systems*, the protocol of Aharonov *et al.* was not originally shown to be secure according to any rigorous cryptographic security definition.

We generalize the protocol of Aharonov *et al.* to support delegated quantum computation (in contrast to only deciding membership in a language) by making two minor modifications. First we instantiate the protocol using any encode-encrypt quantum authentication scheme that admits computing on authenticated data (such as the trap authentication scheme or the signed polynomial scheme as used by Aharonov *et al.*). Analogously to our main protocol, we also introduce as an aid in the proof a control-bit so that the circuit being implemented is a controlled-unitary.

The ideal functionality we achieve is described in Functionality 4. Following [ABOE10], we describe the functionality in terms of a *prover* and *verifier*.

Functionality 4 Ideal functionality $\mathcal{F}_{\Phi}^{\mathrm{delegated}}$ for a quantum channel $\Phi : \mathsf{A} \to \mathsf{C}$.

1. **Create:** Upon input register A from the verifier, send `create` to the prover and store the contents of register A.
2. **Execute:** Upon input `execute` from the prover, evaluate Φ on register A, and send the contents of the output register C to the verifier.

Theorem 11. *Let Φ be polynomial-time quantum computable functionality. Then there exists an efficient, quantum interactive protocol which statistically quantum-UC-emulates $\mathcal{F}_{\Phi}^{\mathrm{delegated}}$ in the case of a corrupt prover, in the plain model, and where the only quantum power required of the verifier is to encode the input and auxiliary quantum registers, and to decode the output. In particular, all the interaction is classical except for the first and last messages.*

The proof of Theorem 11 follows as a special case of our main possibility result. In the case of a general Φ, the registers that the verifier prepares in Theorem 11 are polynomial-size in the security parameter. In the interactive proof scenario of Aharonov *et al.*, the input to Φ is the all-$|0\rangle$ product state, the output is a single classical bit, and it suffices to implement $\mathcal{F}_{\Phi}^{\mathrm{delegated}}$ with only constant security. Given these assumptions, the only quantum power required of the verifier is the ability to prepare constant-sized quantum registers in the first round.

Proof sketch. We view the verifier as the sender in the QOTP, but since the verifier has the input, and receives the output, she can encode and decode these herself, so we do away with the necessity of the encoding and decoding gadgets. Also, classical interaction is permitted, so we replace the BR-OTP with interaction. The simulator and proof are the same. □

Appendices

A One-time programs for classical, bounded reactive functionalities ($\mathcal{F}^{\text{BR-OTP}}$)

Functionality 5 Ideal functionality $\mathcal{F}^{\text{BR-OTP}}_{g_1,\ldots,g_\ell}$ for a bounded, sender-oblivious reactive functionality. Here, $g_1(a, b_1) \mapsto (m_1, s_1)$ and $g_i(b_i, s_{i-1}) \mapsto (m_i, s_i)$ $(i = 2, \ldots, \ell)$ are classical functions; a represents the sender's input, b_i represents the receiver's input for round i (which can depend on the input-output behaviour of previous rounds), s_i represents the internal state after round i, and m_i is the message returned to the receiver after round i. We assume $s_\ell = \perp$.

1. **Create:** Upon input a from the sender, send `create` to the recipient and store a.
2. **Execute:** Upon input (i, b_i) where $i \in \{1, \ldots, \ell\}$ from the recipient, do the following:
   (a) For $j = 1, \ldots, i-1$, if g_j has not been evaluated, abort. If g_i has already been evaluated, abort.
   (b) Compute $g_i(b_i, s_{i-1}) = (m_i, s_i)$ (if $i = 1$, compute $g_1(a, b_1) = (m_1, s_1)$).
   (c) Output m_i to the receiver. Store s_i and note that g_i has been evaluated.
   (d) If m_ℓ has just been output, delete any trace of this instance.

In this section, we use standard techniques to extend Theorem 2 to sender-oblivious, polynomial-time computable, *bounded reactive* classical two-party functionalities (given as Functionality 5). We achieve this result in a straightforward way. In fact, Goyal *et al.* [GIS+10, p. 40] suggest that the result below would follow from their work. This appendix provides all the details, using some of the techniques of Goyal *et al.* [GIS+10].

A *message authentication code (MAC)* is a pair of algorithms (MAC, VF), where $MAC_k(m)$ constructs a tag σ for a message m under a key k, and $VF_k(m, \sigma)$ returns 1 if σ is a valid tag for m under key k. A MAC is an *unconditional one-time secure MAC* if

$$\Pr\left[VF_k(m, \sigma) = 1 : k \xleftarrow{\$} \{0,1\}^\kappa, (m, \sigma) \leftarrow \mathcal{A}()\right] \tag{76}$$

is negligible in a security parameter κ for all probabilistic algorithms $\mathcal{A}$. An example of such a MAC is as follows. The key k is a pair of κ-length binary strings (a, b). The tag for a message $m \in \{0,1\}^\kappa$ is $MAC_{(a,b)}(m) = a \cdot m + b$, where all operations are in $GF(2^\kappa)$.

The basic idea of the construction (see Protocol 6) is to create a one-time program for each next-message function g_i in the ideal functionality $\mathcal{F}^{\text{BR-OTP}}_{g_1,\ldots g_\ell}$. For this, we need a way to guarantee that the receiver executes the COTPs in the correct order, and also let the receiver pass the functionality's state information from one COTP to another without revealing this information to the receiver. Both of these goals can be achieved via a standard authentication and encryption mechanism: we define a family of functions f_i that are based on g_i and output an unconditionally secure authenticated encryption of the sender's internal state at the end of round i; this information should be supplied by the receiver as an additional input for f_{i+1}. If the authentication fails, then f_i outputs $\perp$.

Theorem 12. *Protocol 6 statistically classical-UC emulates $\mathcal{F}^{\text{BR-OTP}}_{g_1,\ldots,g_\ell}$ in the $\mathcal{F}^{\text{COTP}}_f$-hybrid model, assuming MAC is an unconditional one-time secure MAC.*

Protocol 6 Protocol for a bounded, sender-oblivious reactive functionality, $\mathcal{F}_{g_1,\ldots g_\ell}^{\text{BR-OTP}}$ in the $\mathcal{F}_f^{\text{COTP}}$-hybrid model.

1. **Key generation.** For $1 \leq i \leq \ell - 1$, the sender randomly chooses keys k_1^i and k_0^i.
2. **Definition of functions.** For each $1 \leq i \leq \ell$, given g_i, define f_i as follows.
   (a) **Inputs of f_i.**
   - For $i = 1$, function f_1 takes as input from the sender a string a, a key k_1^1 for message authentication scheme $MAC(\cdot)$, and key k_0^1 which has the same length as s_1. Function f_1 takes as input from the receiver a bit-string b_1.
   - For $i > 1$, function f_i takes as input from the sender keys k_1^{i-1} and k_1^i for message authentication scheme $MAC(\cdot)$ and keys k_0^{i-1} and k_0^i which have the same length as s_{i-1} and s_i, respectively. Function f_i takes as input from the receiver bit-strings $(b_i, c_0^{i-1}, c_1^{i-1})$.

   (b) **Computation of f_i.**
   - If $i = 1$, compute $g_1(a, b_1)$ to obtain the message m_1 as well as internal state s_1.
   - If $i > 1$, check that $VF_{{k_1}^{i-1}}(c_0^{i-1}, c_1^{i-1}) = 1$. If not, output $\perp$. Else, set $s_{i-1} = c_0^{i-1} \oplus k_0^{i-1}$. Compute $g_i(b_i, s_{i-1})$ to obtain the i^{th} message m_i as well as internal state s_i.

   (c) **Outputs of f_i.**
   - If $i < \ell$, output $(m_i, s_i \oplus k_0^i, MAC_{k_1^i}(s_i \oplus k_0^i))$.
   - If $i = \ell$, output $(m_i, \perp)$.
3. **COTP construction.** The sender uses $\mathcal{F}_{f_i}^{\text{COTP}}$ to create one-time programs for $f_1, \ldots, f_\ell$.
4. **Evaluation.** The receiver evaluates $g_1, \ldots, g_\ell$ by doing the following:
   - If $i = 1$, run $\mathcal{F}_{f_1}^{\text{COTP}}$ on input b_1 and obtain (m_1, c_0^1, c_1^1).
   - If $i > 1$, run $\mathcal{F}_{f_i}^{\text{COTP}}$ on input $(b_i, c_0^{i-1}, c_1^{i-1})$ and obtain (m_i, c_0^i, c_1^i).

By Theorem 2, there exists a non-interactive protocol ρ that classical-UC-emulates $\mathcal{F}^{\text{COTP}}$ in the $\mathcal{F}^{\text{OTM}}$-hybrid model. Thus we have Corollary 2.1 as given in the main text.

We finish this section with a proof of Theorem 12.

Proof of Theorem 12. The proof proceeds by considering security against either a malicious sender or a malicious receiver.

Security against a malicious sender. Let $\mathcal{A}$ be an adversary corrupting the sender in Protocol 6. We construct a simulator $\text{Sim}_{\mathcal{A}}$ for $\mathcal{A}$.

The simulator is defined as follows: execute $\mathcal{A}$ on input given by $\mathcal{Z}$. This execution determines inputs for $\mathcal{F}_{f_i}^{\text{COTP}}$, and in particular, it determines the sender's input a.

We say that the senders's keys are *consistent* if, for each i, the keys k_1^i and k_0^{i-1} are self-consistent between instantiations of $\mathcal{F}_{f_i}^{\text{COTP}}$ and $\mathcal{F}_{f_{i+1}}^{\text{COTP}}$ (i.e., the ideal functionalities are called with the corresponding same inputs). If the sender's keys are consistent, then an execution of Protocol 6 with an honest receiver will not abort.

Thus, if $\text{Sim}_{\mathcal{A}}$ detects that the sender's keys are *not* consistent, $\text{Sim}_{\mathcal{A}}$ inputs ABRT into the ideal functionality $\mathcal{F}_{g_1,\ldots g_\ell}^{\text{COTP}}$. Otherwise, $\text{Sim}_{\mathcal{A}}$ inputs a for the sender.

We thus have the following:

1. protocol 6 is non-interactive;
2. the probability of ABRT in the ideal model is independent of the receiver's input;
3. an honest sender will not cause the protocol to abort.

Together, these imply that the real and ideal networks are perfectly indistinguishable.

Security against a malicious receiver.

Let $\mathcal{A}$ be an adversary corrupting the receiver in Protocol 6. We construct a simulator $\text{Sim}_{\mathcal{A}}$, given as Simulator 1.

Simulator 1 Simulator $\text{Sim}_{\mathcal{A}}$, for the proof of Protocol 6 against a malicious receiver.

---

1. For $i = 1, \ldots, \ell$, choose keys k_i^0 and k_i^0 as in Protocol 6.
2. Start the execution of $\mathcal{A}$ on the input given by $\mathcal{Z}$; set $i = 1$.
3. Execute $\mathcal{A}$ until a call to the ideal functionality $\mathcal{F}_{f_i}^{\text{COTP}}$ occurs. Do the following in order to simulate interaction with the ideal functionality.
   (a) For $j = 1, \ldots, i-1$, if $\mathcal{F}_{f_j}^{\text{COTP}}$ has not already been evaluated, abort.
   (b) Let σ_i be $\mathcal{A}$'s input into $\mathcal{F}_{f_i}^{\text{COTP}}$.
      i. if $i = 1$, forward σ_1 to $\mathcal{F}_{g_1,\ldots,g_\ell}^{\text{COTP}}$ and receive as response m_1. Choose a random state w, and return $(m_1, w \oplus k_0^1, MAC_{k_1^1}(w \oplus k_0^1))$ to $\mathcal{A}$.
      ii. if $i > 1$, interpret σ_i as $(b_i, c_0^{i-1}, c_1^{i-1})$. Check that $VF_{k_1{}^{i-1}}(c_0^{i-1}, c_1^{i-1}) = 1$. If not, output $\perp$. Forward b_i to $\mathcal{F}_{g_1,\ldots,g_\ell}^{\text{COTP}}$ and receive as response m_i. Choose a random state w_i, and return $(m_i, w_i \oplus k_0^i, MAC_{k_1^i}(w_i \oplus k_0^i))$ to $\mathcal{A}$.
   (c) If $i < \ell$, return to step 3. Otherwise, output $\mathcal{A}$'s output.

The real and ideal networks are indistinguishable: in the real world, the adversary can succeed in out-of-order querying with probability at most the probability of generating a forged MAC, which is negligible since the MAC is secure and $\mathcal{A}$ can succeed in making an OTP accept an incorrect internal state also with negligible probability. □

B Properties of encode-encrypt authentication schemes

B.1 Security against Pauli attacks implies security against general attacks

Parts of this section are reproduced from Aharonov, Ben-Or, and Eban [ABOE10].

B.1.1 How Pauli attacks affect the state of the system

A Pauli attack on a family $\mathscr{E}$ of codes has the following form.

1. The data register D is encoded under a uniformly random choice of code $E \in \mathscr{E}$ by preparing (X,Z) in the $|0\rangle$ state and applying E to $(\mathsf{D},\mathsf{X},\mathsf{Z})$.
2. A malicious attacker applies a Pauli operator Q to $(\mathsf{D},\mathsf{X},\mathsf{Z})$.
3. Data is decoded by applying the inverse operator E^* to $(\mathsf{D},\mathsf{X},\mathsf{Z})$.
4. The syndrome registers (X,Z) are measured in the computational basis. A non-zero measurement result indicates cheating.

The first three steps of this protocol induce a channel Ψ_Q of the form

$$\Psi_Q : \mathsf{D} \to (\mathsf{D},\mathsf{X},\mathsf{Z}) : \rho \mapsto \frac{1}{|\mathscr{E}|}\sum_{E\in\mathscr{E}} E^*QE(\rho\otimes|0\rangle\langle 0|)E^*Q^*E \ .$$

Let $[X],[Z]$ denote the projectors onto the state $|0\rangle$ for syndrome registers X,Z, respectively. The state of the data register D after the syndrome measurement of step 4 is

$$\mathrm{Tr}_{\mathsf{XZ}}\left([X][Z]\Psi_Q(\rho)\right) + \mathrm{Tr}_{\mathsf{XZ}}\left((I-[X][Z])\Psi_Q(\rho)\right) \ . \tag{77}$$

Let us examine the terms associated with each measurement outcome. For each Pauli Q we define the following partition of $\mathscr{E}$:

- $\mathscr{E}_{\mathsf{XZ}}(Q)$: The set of codes $E \in \mathscr{E}$ for which Q acts trivially on logical data and has no error syndrome.
- $\mathscr{E}_{\mathsf{XZ!}}(Q)$: The set of codes $E \in \mathscr{E}$ for which Q acts *non*-trivially on logical data but has no error syndrome.
- $\mathscr{E}_{\mathsf{XZ}\emptyset}(Q)$: The set of codes $E \in \mathscr{E}$ for which Q has non-zero error syndrome.

For each code $E \in \mathscr{E}$ let $Q_{\ell(E)}$ denote the logical Pauli induced by Q, E and observe that if $E \in \mathscr{E}_{\mathsf{XZ}}(Q)$ then $Q_{\ell(E)} = I$. Then

$$\mathrm{Tr}_{\mathsf{XZ}}\left([X][Z]\Psi_Q(\rho)\right) = \underbrace{\frac{1}{|\mathscr{E}|}\sum_{E\in\mathscr{E}_{\mathsf{XZ}}(Q)} Q_{\ell(E)}\rho Q^*_{\ell(E)}}_{\frac{|\mathscr{E}_{\mathsf{XZ}}(Q)|}{|\mathscr{E}|}\rho} + \underbrace{\frac{1}{|\mathscr{E}|}\sum_{E\in\mathscr{E}_{\mathsf{XZ!}}(Q)} Q_{\ell(E)}\rho Q^*_{\ell(E)}}_{[\varepsilon(Q)]} \tag{78}$$

$$\mathrm{Tr}_{\mathsf{XZ}}\left((I-[X][Z])\Psi_Q(\rho)\right) = \frac{1}{|\mathscr{E}|}\sum_{E\in\mathscr{E}_{\mathsf{XZ}\emptyset}(Q)} Q_{\ell(E)}\rho Q^*_{\ell(E)} \ . \tag{79}$$

If $\mathscr{E}$ is ϵ-secure against Pauli attacks (according to Definition 9) then by definition the term marked $[\varepsilon(Q)]$ in (78) has trace at most ϵ and so it must be that $|\mathscr{E}_{\mathsf{XZ!}}(Q)|/|\mathscr{E}| \le \epsilon$ for all Q.

If $Q = I$ then $Q_{\ell(E)} = I$ for all $E \in \mathscr{E}$ and $\mathscr{E}_{\mathsf{XZ}}(Q) = \mathscr{E}$. In this case the terms $[\varepsilon(Q)]$ and (79) both vanish and the state of the system is simply ρ as required for any authentication scheme.

B.1.2 The Pauli sandwich

As mentioned in Section 4.1, encode-encrypt authentication schemes have the desirable property that the Pauli encryption breaks up an arbitrary attack into a probabilistic mixture of Pauli attacks. This property hinges upon a key lemma that we call the *Pauli sandwich*; it is a direct consequence of the *Pauli twirl* [DCEL09]. A succinct proof can be found in Aharonov, Ben-Or, and Eban [ABOE10].

Lemma 13 (Pauli sandwich (see [ABOE10]))**.** *Let W be an arbitrary operator acting on n-qubits and let Q, Q' be Pauli operators acting on n-qubits. It holds that*

$$\frac{1}{4^n} \sum_{Paulis\ P} P^* Q P W P^* Q'^* P = \begin{cases} QWQ^* & \text{if } Q = Q' \\ 0 & \text{otherwise} \end{cases} \tag{80}$$

To see how the Pauli sandwich breaks up an arbitrary attack into a mixture of Pauli attacks let U be any operator acting on a register of n-qubits, let $U = \sum_{\text{Paulis } Q} \alpha_Q Q$ be a decomposition of U as a linear combination of Pauli operators, and consider the channel

$$\Psi : \rho \mapsto \frac{1}{4^n} \sum_{\text{Paulis } P} P^* U P \rho P^* U^* P = \frac{1}{4^n} \sum_{\text{Paulis } P} \sum_{\text{Paulis } Q,\, Q'} \alpha_Q \overline{\alpha_{Q'}} P^* Q P \rho P^* Q'^* P \tag{81}$$

It follows immediately from the Pauli sandwich (Lemma 13) that

$$\Psi : \rho \mapsto \sum_{\text{Paulis } Q} |\alpha_Q|^2 Q \rho Q^* \tag{82}$$

That is, if an adversary applies a unitary attack U to a register encrypted with a uniformly random Pauli P then the resulting state after decryption is indistinguishable from a simplified attack where the adversary instead applies a Pauli attack Q chosen at random according to the distribution $\{|\alpha_Q|^2\}$.

This observation holds for arbitrary operators U (not just unitary operators) so it can easy be extended to arbitrary non-unitary channel attacks Φ by applying a similar analysis for each operator in a Kraus decomposition of Φ.

B.1.3 How general attacks affect the state of the system

An arbitrary attack on an encode-encrypt authentication scheme based on a family $\mathscr{E}$ of codes has the following form.

1. A data register D is authenticated by encoding it with a random code $E \in \mathscr{E}$ (implicitly introducing syndrome registers X, Z) and then encrypting the encoded n-qubit system $(\mathsf{D}, \mathsf{X}, \mathsf{Z})$ with a random Pauli P.
2. A malicious attacker applies a channel Φ to the n qubits comprising the registers $(\mathsf{D}, \mathsf{X}, \mathsf{Z})$.
3. The system is decrypted by applying the Pauli P^* and decoded by applying the inverse circuit E^*.
4. The syndrome registers (X, Z) are measured in the computational basis. A non-zero measurement result indicates cheating.

As mentioned previously, it suffices to restrict attention only to unitary attacks $\Phi : X \mapsto UXU^*$; security against arbitrary channels is recovered by applying the same analysis to each operator in a Kraus decomposition of Φ. The first three steps of this protocol induce a channel Ψ_U of the form

$$\Psi_U : \mathsf{D} \to (\mathsf{D}, \mathsf{X}, \mathsf{Z}) \tag{83}$$

$$\rho \mapsto \frac{1}{4^n |\mathscr{E}|} \sum_{E \in \mathscr{E}} \sum_{\text{Paulis } P} E^* P^* U P E \left(\rho \otimes |0\rangle\langle 0|\right) E^* P^* U^* P E \tag{84}$$

Let $U = \sum_{\text{Paulis } Q} \alpha_Q Q$ be a decomposition of U into a linear combination of Paulis so that the channel Ψ_U could equivalently be written

$$\Psi_U : \rho \mapsto \frac{1}{4^n |\mathscr{E}|} \sum_{E \in \mathscr{E}} \sum_{\text{Paulis } P} \sum_{\text{Paulis } Q, Q'} \alpha_Q \overline{\alpha_{Q'}} E^* P^* Q P E \left(\rho \otimes |0\rangle\langle 0|\right) E^* P^* Q'^* P E \tag{85}$$

$$= \frac{1}{|\mathscr{E}|} \sum_{E \in \mathscr{E}} \sum_{\text{Paulis } Q} |\alpha_Q|^2 E^* Q E \left(\rho \otimes |0\rangle\langle 0|\right) E^* Q^* E \tag{86}$$

$$= \sum_{\text{Paulis } Q} |\alpha_Q|^2 \Psi_Q(\rho) \tag{87}$$

with the first equality following form the Pauli sandwich (Lemma 13).

The state of the register D after the syndrome measurement of step 4 is

$$\mathrm{Tr}_{\mathsf{XZ}} \left([X][Z]\Psi_U(\rho)\right) + \mathrm{Tr}_{\mathsf{XZ}} \left((I - [X][Z])\Psi_U(\rho)\right). \tag{88}$$

As before, let us examine the terms associated with each measurement outcome. We have

$$\mathrm{Tr}_{\mathsf{XZ}} \left([X][Z]\Psi_U(\rho)\right) = \sum_{\text{Paulis } Q} |\alpha_Q|^2 \, \mathrm{Tr}_{\mathsf{XZ}} \left([X][Z]\Psi_Q(\rho)\right) \tag{89}$$

$$= \sum_{\text{Paulis } Q} |\alpha_Q|^2 \left(\frac{|\mathscr{E}_{\mathsf{XZ}}(Q)|}{|\mathscr{E}|}\rho + [\varepsilon(Q)]\right) \tag{90}$$

where the second equality is from (78). From the expression (90) one can see that the probability of acceptance is a nondecreasing function of the modulus squared of α_I. In particular, the more weight U places on the identity in its Pauli decomposition, the better the chance that the attack U is successful (and the least likely it is to have any effect on the state).

For completeness we explicitly write the term associated with outcome $I - [X][Z]$:

$$\mathrm{Tr}_{\mathsf{XZ}} \left((I - [X][Z])\Psi_U(\rho)\right) = \sum_{\text{Paulis } Q} |\alpha_Q|^2 \, \mathrm{Tr}_{\mathsf{XZ}} \left((I - [X][Z])\Psi_Q(\rho)\right) \tag{91}$$

$$= \sum_{\text{Paulis } Q} |\alpha_Q|^2 \left(\frac{1}{|\mathscr{E}|} \sum_{E \in \mathscr{E}_{\mathsf{XZ}\emptyset}(Q)} Q_{\ell(E)} \rho Q^*_{\ell(E)}\right) \tag{92}$$

B.2 Measure-then-decode equals decode-then-measure for CSS codes

CSS codes have the property that measurement of logical data in the computational basis can be implemented by bitwise (transversal) measurement of the physical qubits in the encoding, followed by a purely classical decoding process.

More concretely, suppose E is a CSS code that encodes one logical qubit into n physical qubits. For each value of the bit $a \in \{0,1\}$ there exists a set $D_E(a)$ of n-bit strings such that the encoded logical basis state $E|a\rangle$ is an equal superposition of strings in $D_E(a)$:

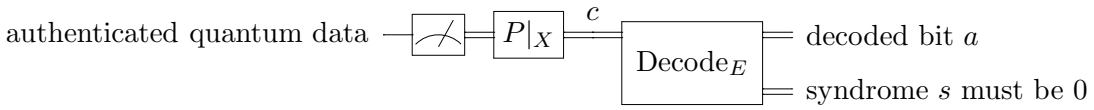
$$E|a\rangle = \frac{1}{\sqrt{|D_E(a)|}} \sum_{b \in D_E(a)} |b\rangle \tag{93}$$

Bitwise measurement of $E|a\rangle$ in the computational basis yields a string $b \in D_E(a)$ selected uniformly at random. The sets $D_E(0), D_E(1)$ are disjoint so the logical measurement result a can be deduced by identifying the set $D_E(a)$ from which b was drawn. Moreover, given an arbitrary n-bit string c there is an efficient classical algorithm that computes the function

$$\text{Decode}_E : c \mapsto (a, s) \tag{94}$$

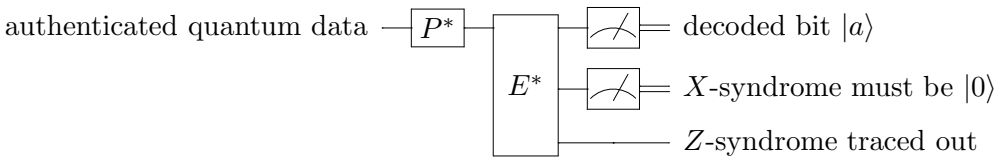
where s is an n-bit syndrome string with the property that $c \oplus s \in D_E(a)$.

Suppose instead that our logical qubit is authenticated under an encode-encrypt scheme based on a family $\mathscr{E}$ of CSS codes and fix a choice of key (E, P) indicating the CSS code E and Pauli encryption P. In this case measurement of the logical qubit can still be implemented by bitwise measurement of the authenticated qubit. The only difference is that the X-portion $P|_X$ of the Pauli key P indicates a mask to be applied before the classical decoding process. This simple process could be drawn as follows.



(95)

The analysis later in this paper is better facilitated by writing this process in a different form. We claim that the above process (95) of measure-then-decode on *classical* data is equivalent to the following process of decode-then-measure on *quantum* data:



(96)

To prove the claim it suffices to show that the decode-then-measure circuit (96) accepts an encrypted n-qubit computational basis state $P|c\rangle$ if and only if $c \in D_E(a)$ for some bit a. It is clear that if $|c\rangle$ does not meet this condition then it will be rejected by (96), so suppose that it does. It is a property of every CSS code that $|c\rangle$ can be written as a superposition of states of the form $QE|a\rangle$ where Q is a purely Z-Pauli. That is,

$$|c\rangle = \sum_{Z\text{-Paulis } Q} \alpha_{c,Q} QE|a\rangle \tag{97}$$

for some complex amplitudes $\{\alpha_{c,Q}\}$. Thus, the encrypted basis state $P|c\rangle$ is accepted by (96) and decodes to $|a\rangle$ as desired.

B.2.1 How general attacks affect the decode-then-measure process

In Section B.1.3 we analyzed the effect of an arbitrary attack on the state of a system protected by an encode-encrypt authentication scheme. In this section we are interested in the effect of such an attack on the decode-then-measure process when the scheme is based on a family $\mathscr{E}$ of CSS codes. The protocol for such an attack is identical to the protocol of Section B.1.3 except that step 4 is replaced with the following.

4. The data register D is measured in the computational basis. The syndrome register X is measured in the computational basis; a non-zero measurement result indicates cheating. The syndrome register Z is discarded.

As usual, it suffices to restrict attention only to unitary channels $\Phi : X \mapsto UXU^*$ for some unitary U with Pauli decomposition $U = \sum_{\text{Paulis } Q} \alpha_Q Q$. The state of the register D after the measurements step 4 is

$$\sum_{\text{Paulis } Q} |\alpha_Q|^2 \sum_{a\in\{0,1\}} |a\rangle\langle a| \left(\sum_{M\in\{[X],I-[X]\}} \text{Tr}_{\mathsf{XZ}} \left(M\Psi_Q(\rho)\right) \right) |a\rangle\langle a|. \tag{98}$$

As before, let us examine the terms associated with each measurement outcome $\{[X], I-[X]\}$. The analysis here is slightly more complicated than in previous sections, owing to the fact that only the X-syndrome is verified in the decode-then-measure protocol. For each Pauli Q we define the following partition of $\mathscr{E}$:

$\mathscr{E}_{\mathsf{X}}(Q)$: The set of codes $E \in \mathscr{E}$ for which $Q_{\ell(E)}$ is a purely Z-Pauli and Q has no X-error syndrome.

$\mathscr{E}_{\mathsf{X}!}(Q)$: The set of codes $E \in \mathscr{E}$ for which $Q_{\ell(E)}$ has nontrivial X-component but Q has no X-error syndrome.

$\mathscr{E}_{\mathsf{X}\emptyset}(Q)$: The set of codes $E \in \mathscr{E}$ for which $Q_{\ell(E)}$ has non-zero X-error syndrome.

Then for each Pauli Q we have

$$\text{Tr}_{\mathsf{XZ}} \left([X]\Psi_Q(\rho)\right) = \frac{1}{|\mathscr{E}|} \sum_{E\in\mathscr{E}_{\mathsf{X}}(Q)} Q_{\ell(E)}\rho Q^*_{\ell(E)} + \underbrace{\frac{1}{|\mathscr{E}|} \sum_{E\in\mathscr{E}_{\mathsf{X}!}(Q)} Q_{\ell(E)}\rho Q^*_{\ell(E)}}_{[\varepsilon_{\mathsf{X}}(Q)]} \tag{99}$$

$$\text{Tr}_{\mathsf{XZ}} \left((I-[X])\Psi_Q(\rho)\right) = \frac{1}{|\mathscr{E}|} \sum_{E\in\mathscr{E}_{\mathsf{X}\emptyset}(Q)} Q_{\ell(E)}\rho Q^*_{\ell(E)} \ . \tag{100}$$

It is a property of every CSS code E that if Q is a purely X- or Z-Pauli then the logical Pauli $Q_{\ell(E)}$ induced by E is also a purely X- or Z-Pauli. Thus, if $\mathscr{E}$ is ϵ-secure against Pauli attacks and if $Q_{\ell(E)}$ has nontrivial X-portion then Q must have nontrivial X-syndrome with probability at most ε taken over the choice of code $E \in \mathscr{E}$. In particular, the term marked $[\varepsilon_{\mathsf{X}}(Q)]$ in (99) has trace at most ϵ and it must be that $|\mathscr{E}_{\mathsf{X}!}(Q)|/|\mathscr{E}| \leq \epsilon$.

On the other hand, if $Q_{\ell(E)}$ is a purely Z-Pauli (which is always the case when $E \in \mathscr{E}_{\mathsf{X}}(Q)$) then

$$|a\rangle\langle a|Q_{\ell(E)}\rho Q^*_{\ell(E)}|a\rangle\langle a| = |a\rangle\langle a|\rho|a\rangle\langle a| \tag{101}$$

for each $a \in \{0,1\}$. It follows that the term in (98) associated with outcome $[X]$ is equal to

$$\sum_{\text{Paulis } Q} |\alpha_Q|^2 \sum_{a\in\{0,1\}} |a\rangle\langle a| \left(\frac{|\mathscr{E}_{\mathsf{X}}(Q)|}{|\mathscr{E}|}\rho + [\varepsilon_{\mathsf{X}}(Q)] \right) |a\rangle\langle a|. \tag{102}$$

Moreover, if Q is a purely Z-Pauli then for every $E \in \mathscr{E}$ it holds that $Q_{\ell(E)}$ is also a purely Z-Pauli and that Q has no X-syndrome under E. That is, $\mathscr{E}_{\mathsf{X}}(Q) = \mathscr{E}$ and the terms $[\varepsilon_{\mathsf{X}}(Q)]$ and (100) both vanish. Thus, one can see from the expression (102) that the probability of acceptance is a nondecreasing function of $\sum_{Z\text{-Paulis } Q} |\alpha_Q|^2$. In particular, the more weight U places on purely Z-Paulis in its Pauli decomposition, the better the chance that the attack U is successful (and the least likely it is to have any effect on the measured state).

For completeness we explicitly write the term in (98) associated with outcome $I - [X]$:

$$\sum_{\text{Paulis } Q} |\alpha_Q|^2 \sum_{a\in\{0,1\}} |a\rangle\langle a| \left(\frac{1}{|\mathscr{E}|} \sum_{E\in\mathscr{E}_{\mathsf{X}\emptyset}(Q)} Q_{\ell(E)}\rho Q^*_{\ell(E)} \right) |a\rangle\langle a|. \tag{103}$$

C Analysis of teleportation

Suppose that a pair of n-qubit registers $(\mathsf{In}, \mathsf{Out})$ is prepared in a "teleport-through-C" state for some n-qubit unitary C. Such a state is constructed by preparing n copies of the entangled state $(|00\rangle + |11\rangle)/\sqrt{2}$ in the registers $(\mathsf{In}, \mathsf{Out})$ and then applying C to register Out. Suppose further that an n-qubit register D is prepared in an arbitrary pure state $|\psi\rangle$ with the intention that this state be teleported through C to qubit Out by way of a standard Bell measurement on $(\mathsf{D}, \mathsf{In})$.

The Bell measurement is implemented in the usual way by applying an n-fold Bell rotation B (each composed of CNOT and H gates) to the registers $(\mathsf{D}, \mathsf{In})$ followed by a measurement of those two registers in the computational basis. After this measurement the registers $(\mathsf{D}, \mathsf{In})$ are in the classical basis state $|T\rangle$ indicating a uniformly random Pauli correction T. Conditioned on this measurement outcome, it is easily seen that the pure state of register Out is $CT|\psi\rangle$.

In the basic teleportation protocol C is the identity and so the state $|\psi\rangle$ can be recovered by applying the appropriate Pauli correction T, thus "teleporting" the state $|\psi\rangle$ from register D to register Out. If C is a Clifford circuit then the state $C|\psi\rangle$ can be recovered by applying an appropriate Pauli correction T_C, thus "teleporting" the state $|\psi\rangle$ from register D "through C" and into register Out.

C.1 Teleportation under attack

Now suppose that all three registers $(\mathsf{D}, \mathsf{In}, \mathsf{Out})$ are passed through a channel Φ prior to the Bell measurement. One could think of Φ as noise, or perhaps a malicious attack on the registers. How does Φ affect the teleportation? In particular, what is the state of the system after the Bell measurement is complete?

Let's start with the special case where Φ is a product unitary $U = U_{\mathsf{D}} \otimes U_{\mathsf{In}} \otimes U_{\mathsf{Out}}$. In this case the pure state of the entire system after the Bell rotation is easily seen to be

$$\frac{1}{2^{n/2}} \sum_{\text{Paulis } T} |T\rangle \otimes U_{\mathsf{Out}} C U_{\mathsf{In}}^{\mathsf{T}} T U_{\mathsf{D}} |\psi\rangle. \tag{104}$$

Measurement of $(\mathsf{D}, \mathsf{In})$ then yields a uniformly random outcome T leaving register Out in the pure state

$$U_{\mathsf{Out}} C U_{\mathsf{In}}^{\mathsf{T}} T U_{\mathsf{D}} |\psi\rangle. \tag{105}$$

The mapping (104) is linear in each of $U_{\mathsf{D}}, U_{\mathsf{In}}, U_{\mathsf{Out}}$ so it can be used to deduce the effect of arbitrary, possibly non-product unitaries U on the state of the registers. This is accomplished by decomposing U into a linear combination of product unitaries and then applying the above identity to each term in that decomposition. For example, let

$$U = \sum_{\text{Paulis } P} \alpha_P P \tag{106}$$

be a decomposition of U into a linear combination of Pauli operators of the form $P = P_{\mathsf{D}} \otimes P_{\mathsf{In}} \otimes P_{\mathsf{Out}}$. By the above analysis it holds that the pure state of the entire system after the Bell rotation is given by

$$\frac{1}{2^{n/2}} \sum_{\text{Paulis } T} |T\rangle \otimes \left(\sum_{\text{Paulis } P} \alpha_P P_{\mathsf{Out}} C P_{\mathsf{In}}^{\mathsf{T}} T P_{\mathsf{D}} \right) |\psi\rangle. \tag{107}$$

Measurement of $(\mathsf{D}, \mathsf{In})$ yields an outcome T leaving register Out in the unnormalized pure state

$$\left(\sum_{\text{Paulis } P} \alpha_P P_{\mathsf{Out}} C P_{\mathsf{In}}^{\mathsf{T}} T P_{\mathsf{D}} \right) |\psi\rangle. \tag{108}$$

The distribution of measurement outcomes T obtained by measurement of $(\mathsf{D}, \mathsf{In})$ need not be uniform, owing to the potential for interference in the sum over amplitudes α_P.

This analysis applies even to non-unitary operators U. Thus, a similar expression can be derived for arbitrary, possibly non-unitary channels Φ by applying the above analysis to each individual Kraus operator in a Kraus decomposition of Φ.

C.2 Teleportation under attack, tabular analysis

It is a useful exercise to repeat the analysis of the previous section in light of the tabular notation introduced in Section 6.5. Let us recall the specification of the attack on teleportation:

1. Registers $(\mathsf{In}, \mathsf{Out})$ are prepared in the pure state $C|\phi^+\rangle$ where $|\phi^+\rangle$ is shorthand for n EPR pairs.
2. An arbitrary attack unitary U is applied to $(\mathsf{D}, \mathsf{In}, \mathsf{Out})$.
3. A Bell measurement is applied to $(\mathsf{D}, \mathsf{In})$ by via Bell rotation B followed by a measurement in the computational basis, which is denoted $\{|T\rangle\langle T|\}$.

The entire procedure may be viewed as a channel

$$\Phi : \mathsf{D} \to (\mathsf{D}, \mathsf{In}, \mathsf{Out}) \tag{109}$$

$$\rho \mapsto \sum_{\text{Paulis } T} |T\rangle\langle T| BUC \left(\rho \otimes |\phi^+\rangle\langle\phi^+| \right) C^* U^* B^* |T\rangle\langle T| \ . \tag{110}$$

For each fixed Pauli T consider the Kraus operator

$$|T\rangle\langle T| BUC |\phi^+\rangle \tag{111}$$

belonging to the channel Φ. This Kraus operator is represented by the following table.

D	$\|T\rangle\langle T\|$	B	U		
In					$\|\phi^+\rangle$
Out				C	

(112)

Substituting the Pauli-decomposition (106) of U into the Kraus operator (111) yields

$$(111) = \sum_{\text{Paulis } P} \alpha_P |T\rangle\langle T|BPC|\phi^+\rangle \ . \tag{113}$$

Fix a choice of Pauli $P = P_{\mathsf{D}} \otimes P_{\mathsf{In}} \otimes P_{\mathsf{Out}}$ and consider the operator $|T\rangle\langle T|BPC|\phi^+\rangle$. This operator can be written in tabular form

D	$\|T\rangle\langle T\|$	B	P_{D}		
In			P_{In}		$\|\phi^+\rangle$
Out			P_{Out}	C	

(114)

so that

$$\boxed{(112)} = \sum_{\text{Paulis } P} \alpha_P \boxed{(114)} \tag{115}$$

The observations of the previous sections can be phrased as an identity between tables:

$$\boxed{(114)} = \begin{array}{|c||c|c|c|c|c|c|} \hline \mathsf{D} & & & & & & \frac{1}{2^{n/2}}|T\rangle \\ \hline \mathsf{In} & & & & & & \\ \hline \mathsf{Out} & P_{\mathsf{Out}} & C & P_{\mathsf{In}}^{\mathsf{T}} & T & P_{\mathsf{D}} & \\ \hline \end{array} \tag{116}$$

Often it is more convenient to simply remove the rows $(\mathsf{D}, \mathsf{In})$ and write

Out	P_{Out}	C	$P_{\mathsf{In}}^{\mathsf{T}}$	T	P_{D}

(117)

so that

$$\boxed{(112)} = \frac{1}{2^{n/2}}|T\rangle \otimes \sum_{\text{Paulis } P} \alpha_P \boxed{(114)} \tag{118}$$

Acknowledgements

We gratefully acknowledge helpful discussions with Daniel Gottesman, Harry Buhrman, Christian Schaffner, Bruce Richmond and Dominique Unruh. A.B. acknowledges support from the Canadian Institute for Advanced Research (CIFAR), Canada's NSERC and Industry Canada. G.G. acknowledges support from Industry Canada, Ontario's Ministry of Research and Innovation, NSERC, DTO-ARO, CIFAR, and QuantumWorks. Part of this research conducted while D.S. was a visitor at the University of Waterloo's IQC.

References

[Aar09] Scott Aaronson. Quantum copy-protection and quantum money. In *Proc. 24th IEEE Conference on Computational Complexity (CCC) 2009*, pp. 229–242, 2009. DOI:10.1109/CCC.2009.42.

[ABOE10] Dorit Aharonov, Michael Ben-Or, and Elad Eban. Interactive proofs for quantum computations. In *Proc. Innovations in Computer Science (ICS) 2010*, pp. 453–469, 2010. EPRINT arXiv:0810.5375v2, URL http://conference.itcs.tsinghua.edu.cn/ICS2010/content/papers/35.html.

[AC12] Scott Aaronson and Paul Christiano. Quantum money from hidden subspaces. In *Proc. 44th Symposium on Theory of Computing (STOC) 2012*, pp. 41–60, 2012. DOI:10.1145/2213977.2213983. Full version availables as arXiv:1203.4740.

[BB84] Charles H. Bennett and Gilles Brassard. Quantum cryptography: Public key distribution and coin tossing. In *Proc. International Conf. on Computers, Systems and Signal Processing*, pp. 175–179. Bangalore, India, 1984.

[BCG+02] Howard Barnum, Claude Crépeau, Daniel Gottesman, Adam Smith, and Alain Tapp. Authentication of quantum messages. In *Proc. 43rd IEEE Symposium on Foundations of Computer Science (FOCS) 2002*, pp. 449–458, 2002. DOI:10.1109/SFCS.2002.1181969. Full version available as arXiv:quant-ph/0205128.

[BCS12] Harry Buhrman, Matthias Christandl, and Christian Schaffner. Complete insecurity of quantum protocols for classical two-party computation. *Physical Review Letters*, **109**:160501, Oct 2012. DOI:10.1103/PhysRevLett.109.160501.

[BFGH10] Debajyoti Bera, Stephen Fenner, Frederic Green, and Steve Homer. Efficient universal quantum circuits. *Quantum Information and Computation*, **10**(1):16–28, January 2010. EPRINT arXiv:0804.2429, URL http://www.rintonpress.com/journals/qiconline.html#v10n12.

[BFK09] Anne Broadbent, Joseph Fitzsimons, and Elham Kashefi. Universal blind quantum computation. In *Proc. 50th IEEE Symposium on Foundations of Computer Science (FOCS) 2009*, pp. 517–526. IEEE, 2009. DOI:10.1109/FOCS.2009.36. EPRINT arXiv:0807.4154.

[BGI+01] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In *Advances in Cryptology – Proc. CRYPTO 2001*, *LNCS*, volume 2139, pp. 1–18. Springer, 2001. DOI:10.1007/3-540-44647-8_1. Full version available as http://www.wisdom.weizmann.ac.il/~oded/p_obfuscate.html.

[BMP+00] P. Oscar Boykin, Tal Mor, Matthew Pulver, Vwani Roychowdhury, and Farrokh Vatan. On universal and fault-tolerant quantum computing. *Information Processing Letters*, **75**:101–107, 2000. EPRINT arXiv:quant-ph/9906054v1.

[BOCG+06] Michael Ben-Or, Claude Crépeau, Daniel Gottesman, Avinatan Hassidim, and Adam Smith. Secure multiparty quantum computation with (only) a strict honest majority. In *Proc. 47th IEEE Symposium on Foundations of Computer Science (FOCS) 2006*, pp. 249–260, 2006. DOI:10.1109/FOCS.2006.68. EPRINT arXiv:0801.1544v1.

[BOM04] Michael Ben Or and Dominic Mayers. General security definition and composability for quantum & classical protocols. arXiv:quant-ph/0409062, 2004.

[Can01] Ran Canetti. Universally composable security: a new paradigm for cryptographic protocols. In *Proc. 42nd IEEE Symposium on Foundations of Computer Science (FOCS) 2001*, pp. 136–145, 2001. DOI:10.1109/SFCS.2001.959888. Updated version available at http://eprint.iacr.org/2000/067.

[DCEL09] Christoph Dankert, Richard Cleve, Joseph Emerson, and Etera Livine. Exact and approximate unitary 2-designs and their application to fidelity estimation. *Physical Review A*, **80**(1):012304, 2009. DOI:10.1103/PhysRevA.80.012304.

[DN06] Christopher Dawson and Michael Nielsen. The Solovay–Kitaev algorithm. *Quantum Information and Computation*, **6**(1):81–95, 2006. EPRINT arXiv:quant-ph/0505030, URL http://www.rintonpress.com/journals/qiconline.html#v6n1.

[DNS12] Frédéric Dupuis, Jesper Buus Nielsen, and Louis Salvail. Actively secure two-party evaluation of any quantum operation. In *Advances in Cryptology – Proc. CRYPTO 2012*, *LNCS*, volume 7417, pp. 794–811. Springer, 2012. DOI:10.1007/978-3-642-32009-5_46.

[dSR07] Paulo Benício de Sousa and Rubens Viana Ramos. Universal quantum circuit for N-qubit quantum gate: a programmable quantum gate. *Quantum Information and Computation*, **7**(3):228–242, March 2007. EPRINT arXiv:quant-ph/0602174, URL http://www.rintonpress.com/journals/qiconline.html#v7n3.

[GC99] Daniel Gottesman and Issac Chuang. Demonstrating the viability of universal quantum computation using teleportation and single-qubit operations. *Nature*, **402**:390–393, 1999. DOI:10.1038/46503. arXiv:quant-ph/9908010v1.

[GIS+10] Vipul Goyal, Yuval Ishai, Amit Sahai, Ramarathnam Venkatesan, and Akshay Wadia. Founding cryptography on tamper-proof hardware tokens. In *Proc. Theory of Cryptography Conference (TCC) 2010*, *LNCS*, volume 5978, pp. 308–326. Springer, 2010. DOI:10.1007/978-3-642-11799-2_19. Full version available at http://eprint.iacr.org/2010/153.

[GKR08] Shafi Goldwasser, Yael Kalai, and Guy Rothblum. One-time programs. In *Advances in Cryptology – Proc. CRYPTO 2008*, *LNCS*, volume 5157, pp. 39–56. Springer, 2008. DOI:10.1007/978-3-540-85174-5_3.

[MS10] Michele Mosca and Douglas Stebila. Quantum coins. In *Error-Correcting Codes, Finite Geometries and Cryptography*, *Contemporary Mathematics*, volume 523, pp. 35–47. American Mathematical Society, 2010. EPRINT arXiv:0911.1295.

[NC97] Michael A. Nielsen and Isaac L. Chuang. Programmable quantum gate arrays. *Physical Review Letters*, **79**:321–324, 1997. DOI:10.1103/PhysRevLett.79.321.

[PW01] Birgit Pfitzmann and Michael Waidner. A model for asynchronous reactive systems and its application to secure message transmission. In *Proc. 22nd IEEE Symposium on Security & Privacy (S&P) 2001*, pp. 184–200. IEEE, 2001. DOI:10.1109/SECPRI.2001.924298. Full version available at http://eprint.iacr.org/2000/066.

[Smi81] Miles E. Smid. Integrating the Data Encryption Standard into computer networks. *IEEE Transactions on Communications*, **29**(6):762–772, June 1981. DOI:10.1109/TCOM.1981.1095071.

[SP00] Peter Shor and John Preskill. Simple proof of security of the BB84 quantum key distribution protocol. *Physical Review Letters*, **85**:441–444, 2000. DOI:10.1103/PhysRevLett.85.441.

[Unr04] Dominique Unruh. Simulatable security for quantum protocols. arXiv:quant-ph/0409125, 2004.

[Unr10] Dominique Unruh. Universally composable quantum multi-party computation. In *Advances in Cryptology – Proc. EUROCRYPT 2010*, *LNCS*, volume 6110, pp. 486–505. Springer, 2010. DOI:10.1007/978-3-642-13190-5_25. Full version availables as arXiv:0910.2912.

[Wie83] Stephen Wiesner. Conjugate coding. *ACM SIGACT News*, **15**(1):78–88, 1983. DOI:10.1145/1008908.1008920. Original article written circa 1970.

[WZ82] William K. Wootters and Wojciech H. Zurek. A single quantum cannot be cloned. *Nature*, **299**(5886):802–803, 1982. DOI:10.1038/299802a0.